(12) UK Patent Application (19) GB (11) 2 230 627 (13) A

(43) Date of A publication 24.10.1990

(21) Application No 8913525.5

(22) Date of filing 13.06.1989

(62) Derived from Application No 8808025.4 under Section 15(4) of the Patents Act 1977

(51) INT CL⁵
G06F 7/49

(52) UK CL (Edition K)
G4A AMD

(56) Documents cited
None

(58) Field of search
UK CL (Edition J) G4A AMD
INT CL⁴ G06F

(71) Applicant
The Secretary of State for Defence
Whitehall, London, SW1A 2HB, United Kingdom

(72) Inventors
Simon Christian Knowles
John Graham McWhirter
John Vincent McCanny
Roger Francis Woods

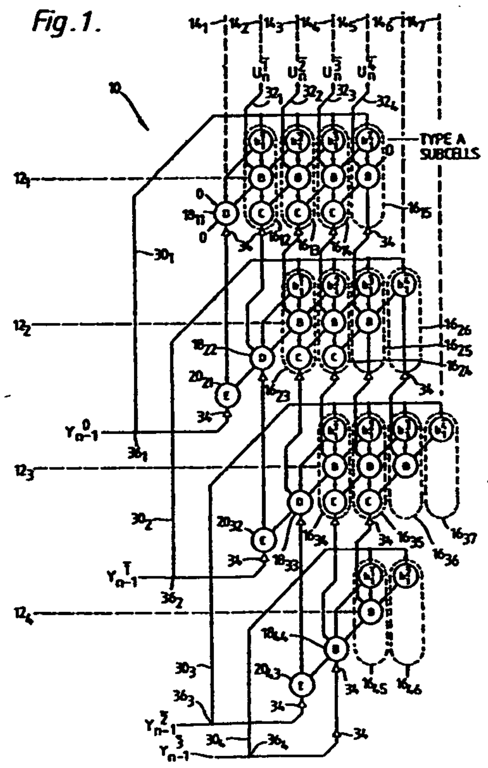
(74) Agent and/or Address for Service
R W Beckham
Procurement Executive, Ministry of Defence,
Patents 1A (4), Room 2014, Empress State Building,
Lillie Road, London, SW6 1TR, United Kingdom

(54) Recursive processor for multiplication

(57) A recursive processor suitable for infinite impulse response (IIR) filter applications incorporates multiplier cells 16 connected to form rows 12 and columns 14. Each row is arranged to multiply by a coefficient. It begins with accumulator cells 8 and 20, and continues with multiplier cells 16 arranged to multiply by individual coefficient digits and disposed in descending order of digit significance. Columns 14 other than the first column 14_1 begin with a multiplier cell 16, and the higher significance columns 14_1 to 14_4 terminate at respective accumulator cells 20. Any intervening multiplier cells are arranged in ascending order of multiplier digit significance. The multiplier and accumulator cells 16 to 20 operate in accordance with signed digit number representation arithmetic involving digit redundancy. They generate sum and transfer digits for output down columns and along rows respectively to neighbouring cells in the direction of increasing coefficient digit significance. The cell arithmetic employed makes it possible to compute results most significant digit first. Each result digit is recycled when formed to provide a multiplicand digit input for all multiplier cells of a respective row selected in accordance with result digit significance. The processor is also arranged to add successive non-recursive input terms to multiplier products. This provides for the processor to provide a first order IIR filter section, and two cascaded processors provide a second order IIR filter section.

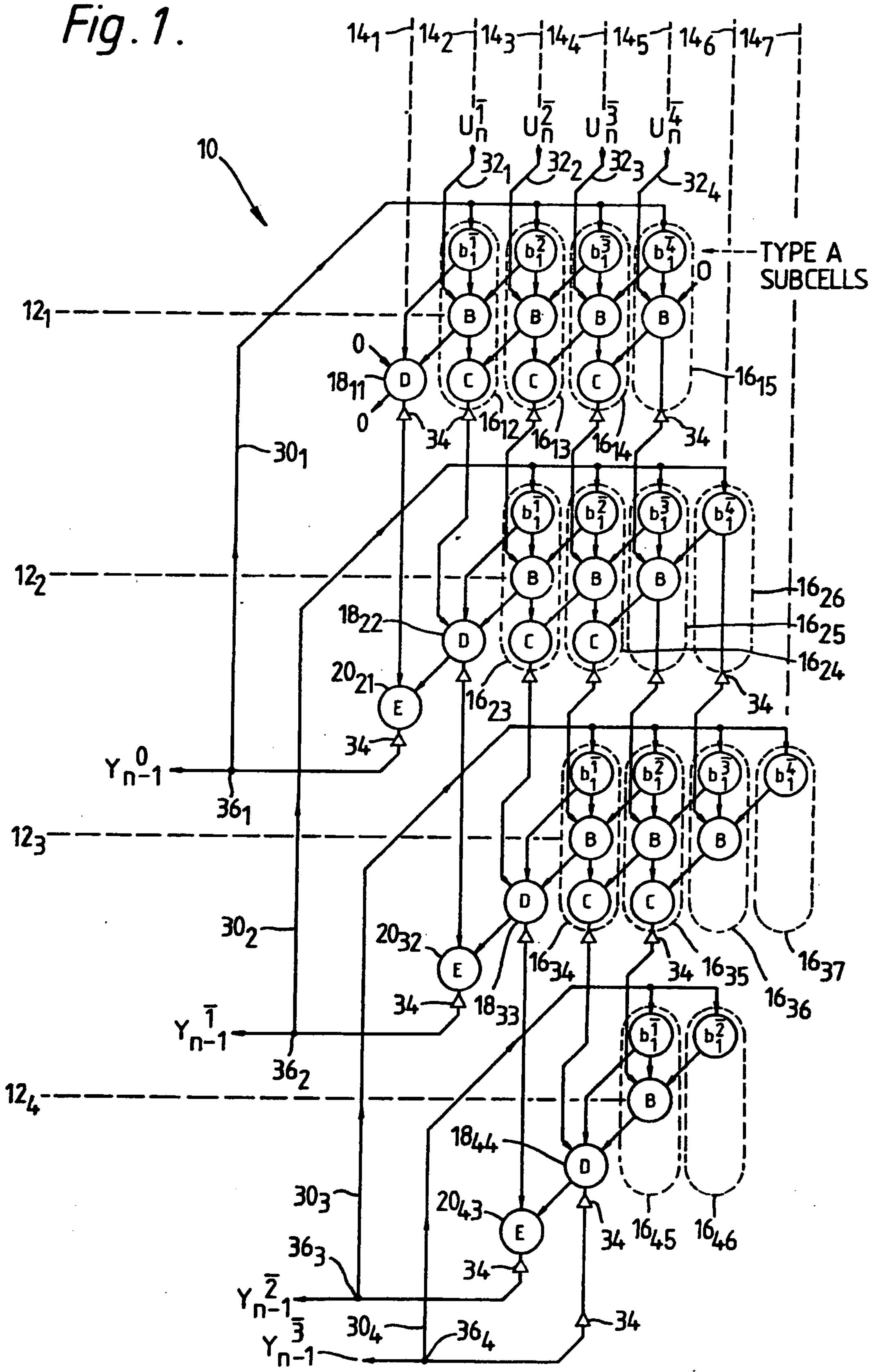
The recursive computations required for a first order IIR filter section is effected irrespective of word length with latency of only two clock cycles.

Fig. 1.



At least one drawing originally filed was informal and the print reproduced here is taken from a later filed formal copy.

Fig. 1.



10
14_1 14_2 14_3 14_4 14_5 14_6 14_7
$U_n^{\bar{1}}$ $U_n^{\bar{2}}$ $U_n^{\bar{3}}$ $U_n^{\bar{4}}$
32_1 32_2 32_3 32_4
TYPE A SUBCELLS
12_1 12_2 12_3 12_4
16_{12} 16_{13} 16_{14} 16_{15}
16_{23} 16_{24} 16_{25} 16_{26}
16_{34} 16_{35} 16_{36} 16_{37}
16_{45} 16_{46}
18_{11} 18_{22} 18_{33} 18_{44}
20_{21} 20_{32} 20_{43}
30_1 30_2 30_3 30_4
36_1 36_2 36_3 36_4
34
0
Y_{n-1}^{0}
$Y_{n-1}^{\bar{1}}$
$Y_{n-1}^{\bar{2}}$
$Y_{n-1}^{\bar{3}}$

Fig.2.

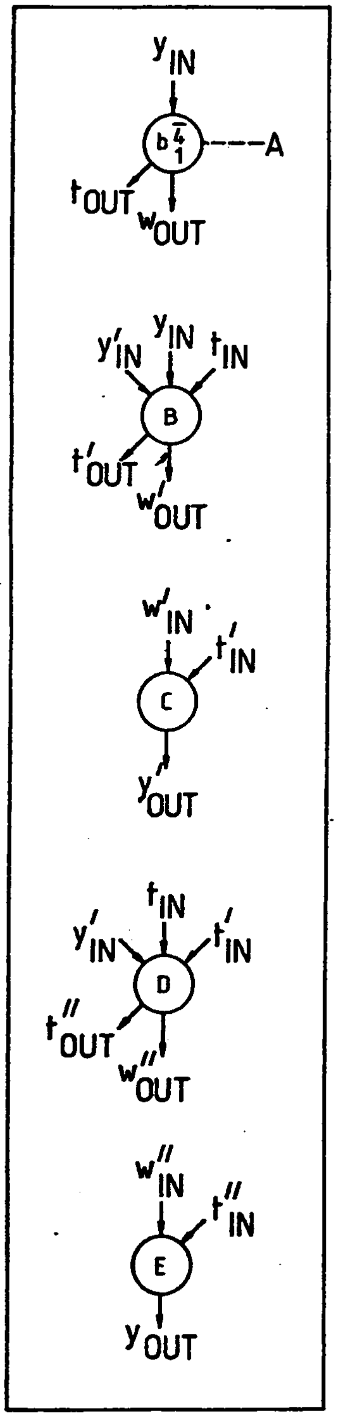

y_{IN}
$b_1^{\bar{4}}$
A
t_{OUT}
w_{OUT}
y'_{IN}
y_{IN}
t_{IN}
B
t'_{OUT}
w'_{OUT}
w'_{IN}
t'_{IN}
C
y'_{OUT}
y'_{IN}
t_{IN}
t'_{IN}
D
t''_{OUT}
w''_{OUT}
w''_{IN}
t''_{IN}
E
y_{OUT}

Fig.3A.

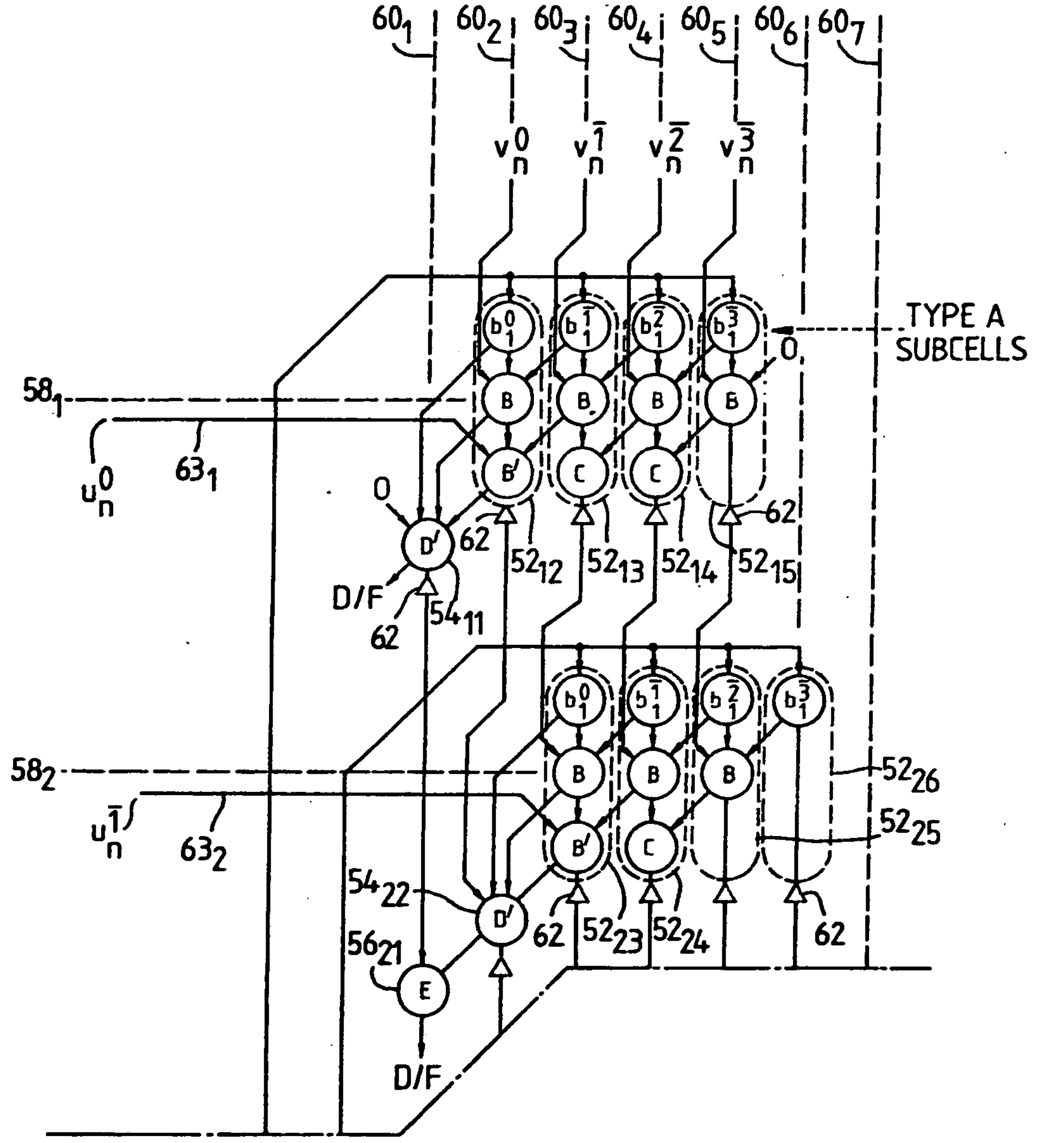
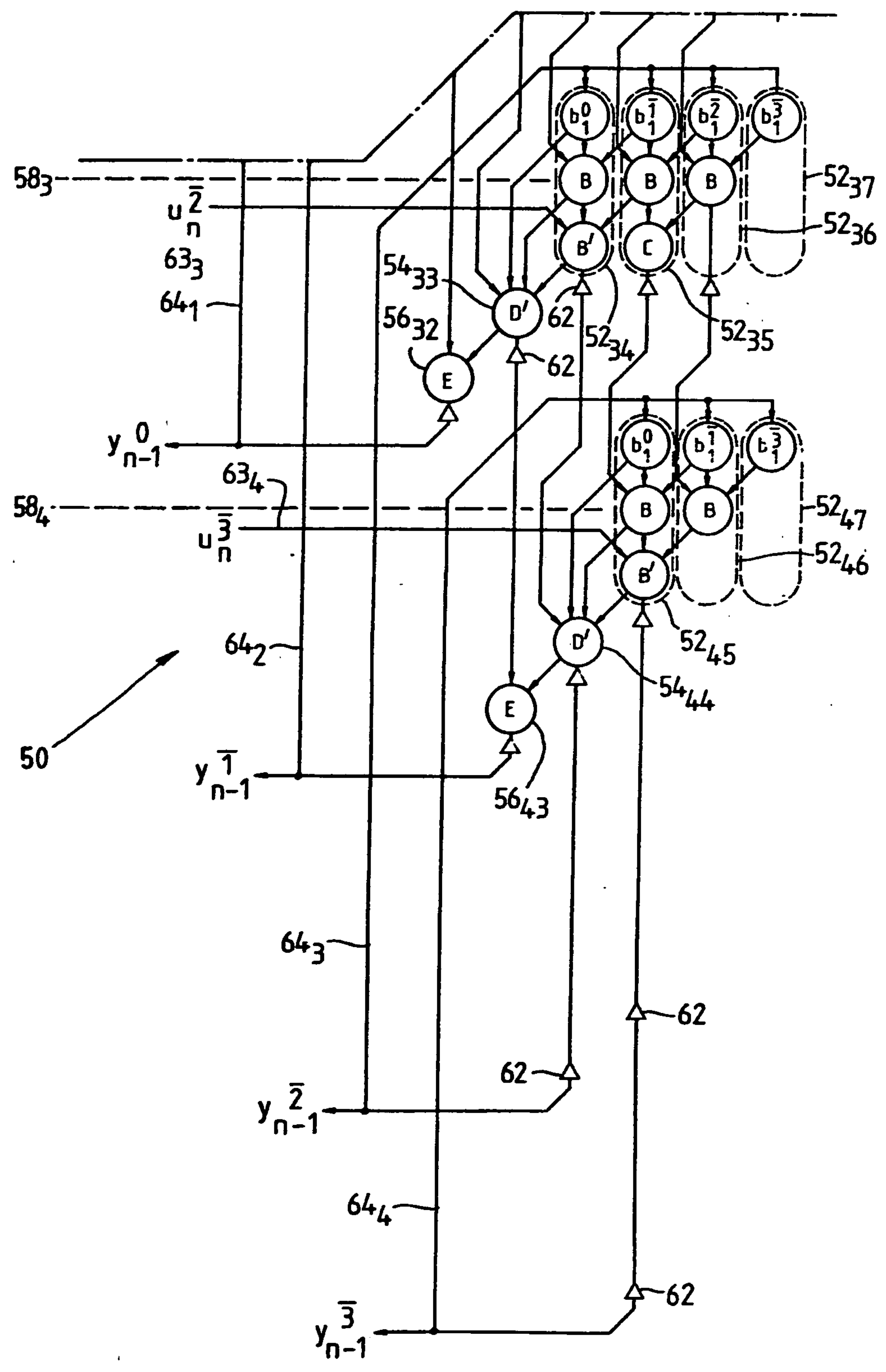
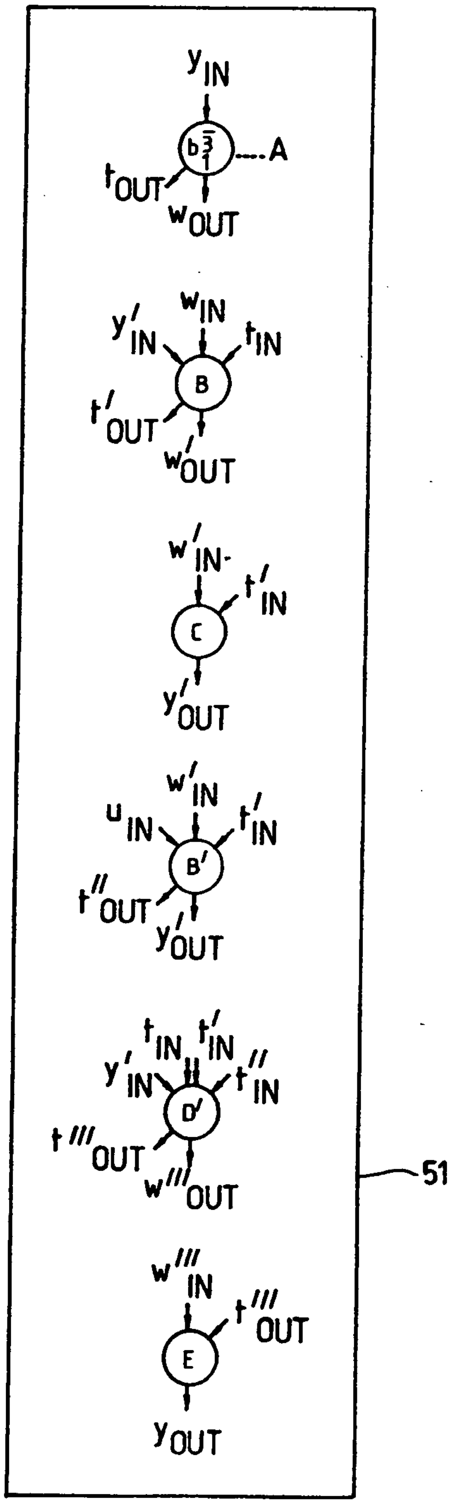

60_1
60_2
60_3
60_4
60_5
60_6
60_7
v_n^0
$v_n^{\bar{1}}$
$v_n^{\bar{2}}$
$v_n^{\bar{3}}$
TYPE A SUBCELLS
b_1^0
$b_1^{\bar{1}}$
$b_1^{\bar{2}}$
$b_1^{\bar{3}}$
0
B
B'
C
D'
E
58_1
58_2
63_1
63_2
u_n^0
$u_n^{\bar{1}}$
62
52_{12}
52_{13}
52_{14}
52_{15}
52_{23}
52_{24}
52_{25}
52_{26}
54_{11}
54_{22}
56_{21}
D/F

Fig. 3B.



b_1^0
$b_1^{\bar{1}}$
$b_1^{\bar{2}}$
$b_1^{\bar{3}}$
B
B'
C
D'
E
58_3
$u_n^{\bar{2}}$
63_3
64_1
54_{33}
56_{32}
62
52_{34}
52_{35}
52_{36}
52_{37}
y_{n-1}^0
63_4
58_4
$u_n^{\bar{3}}$
52_{47}
52_{46}
52_{45}
54_{44}
64_2
50
$y_{n-1}^{\bar{1}}$
56_{43}
64_3
$y_{n-1}^{\bar{2}}$
64_4
$y_{n-1}^{\bar{3}}$

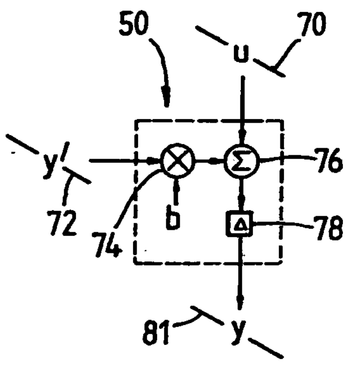
Fig. 3C.



y_{IN}
$b_1^{\bar{3}}$
A
t_{OUT}
w_{OUT}
y'_{IN}
w_{IN}
t_{IN}
B
t'_{OUT}
w'_{OUT}
w'_{IN}
t'_{IN}
C
y'_{OUT}
u_{IN}
w'_{IN}
t'_{IN}
B'
t''_{OUT}
y'_{OUT}
t_{IN}
t'_{IN}
y'_{IN}
t''_{IN}
D'
t'''_{OUT}
w'''_{OUT}
51
w'''_{IN}
t'''_{OUT}
E
y_{OUT}

first order IIR

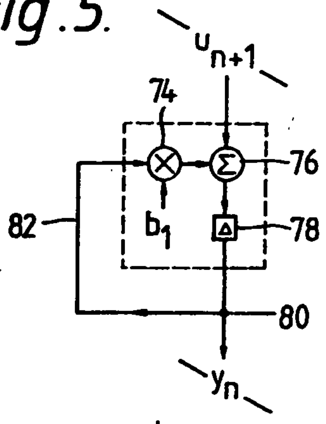
Fig.4.



$y = by' + u$

BASIC BUILDING BLOCK

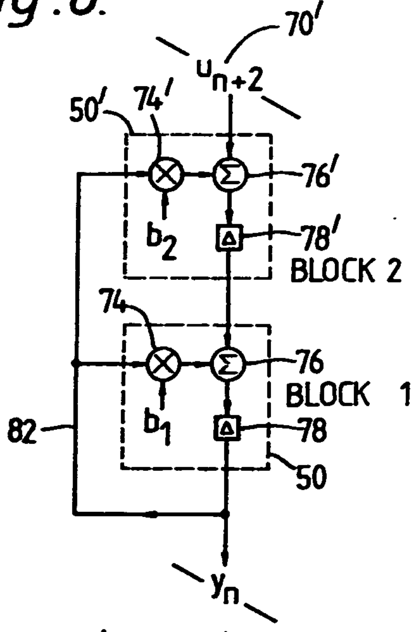
Fig.5.



$y_n = b_1 y_{n-1} + u_n$

FIRST ORDER CONNECTION

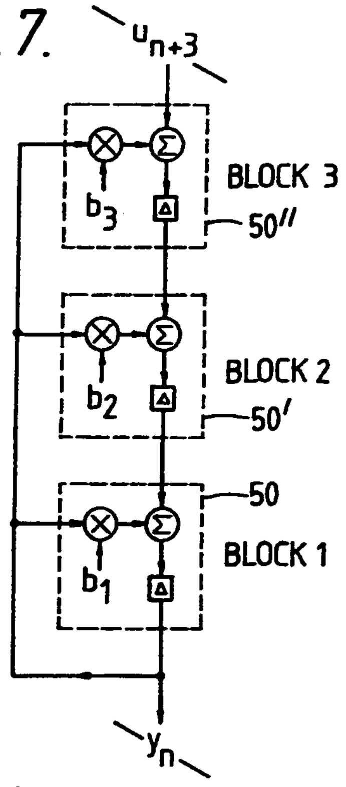
Fig.6.



$y_n = b_1 y_{n-1} + b_2 y_{n-2} + u_n$

SECOND ORDER CONNECTION

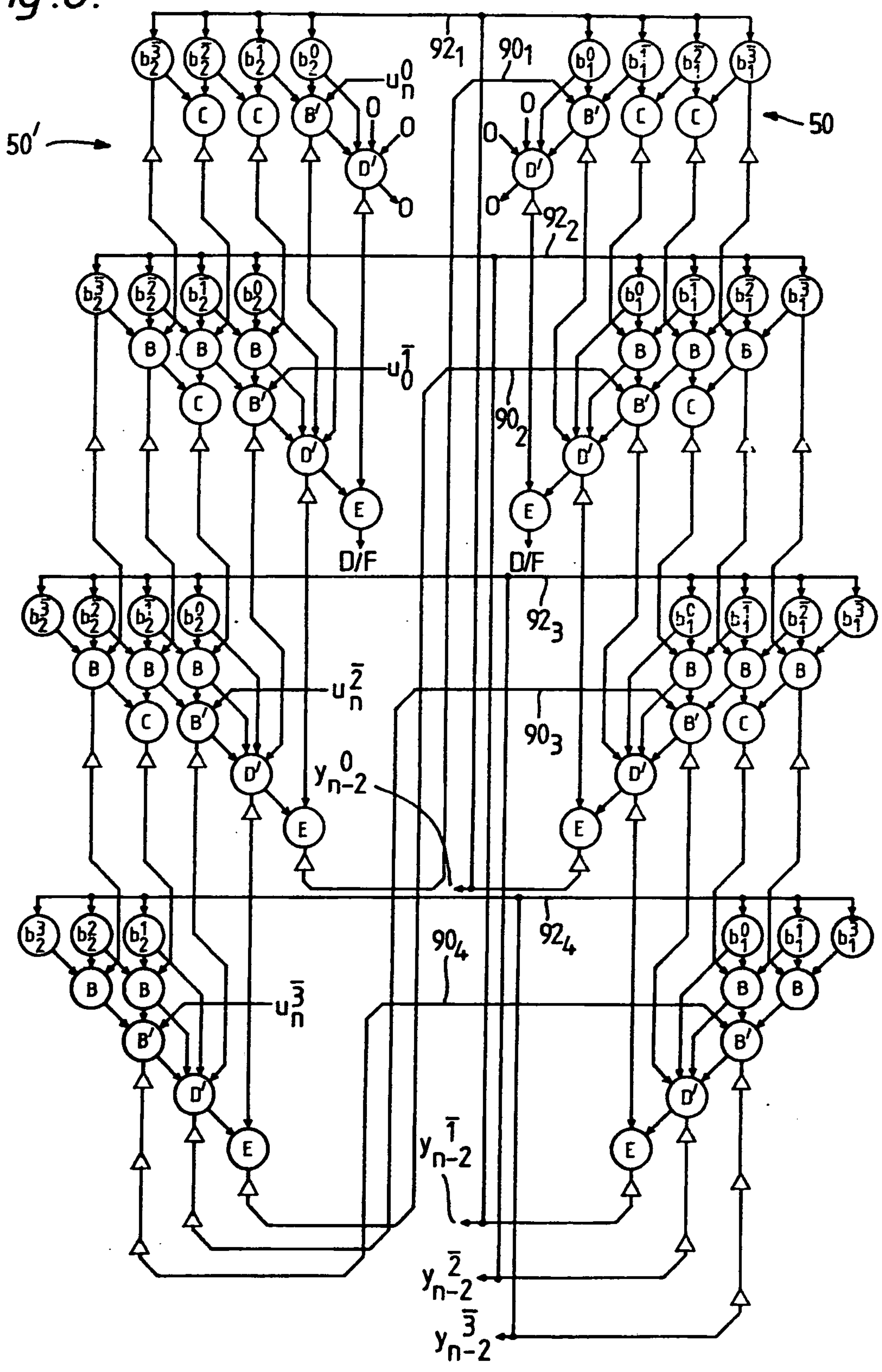
Fig.7.



$y_n = b_1 y_{n-1} + b_2 y_{n-3} + b_3 y_{n-3} + u_n$

THIRD ORDER CONNECTION

Fig.8.



50'
50
92_1
90_1
92_2
90_2
92_3
90_3
92_4
90_4
u_n^0
$u_0^{\bar{1}}$
$u_n^{\bar{2}}$
$u_n^{\bar{3}}$
D/F
D/F
y_{n-2}^0
$y_{n-2}^{\bar{1}}$
$y_{n-2}^{\bar{2}}$
$y_{n-2}^{\bar{3}}$

Fig.9.

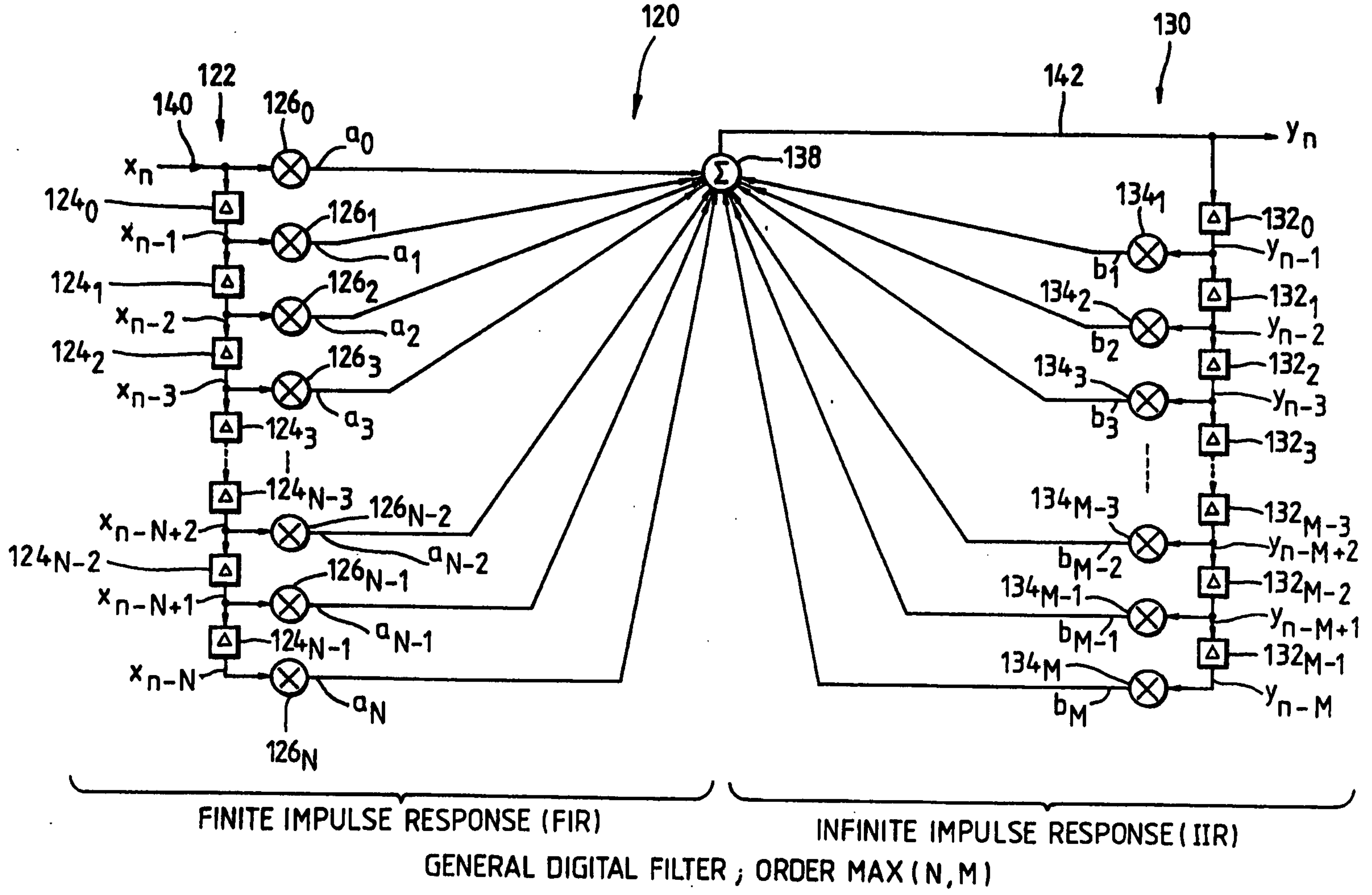

120
130
142
122
140
x_n
y_n
138
126_0
a_0
124_0
x_{n-1}
126_1
a_1
124_1
x_{n-2}
126_2
a_2
124_2
x_{n-3}
126_3
a_3
124_3
124_{N-3}
x_{n-N+2}
126_{N-2}
a_{N-2}
124_{N-2}
x_{n-N+1}
126_{N-1}
a_{N-1}
124_{N-1}
x_{n-N}
a_N
126_N
134_1
b_1
132_0
y_{n-1}
134_2
b_2
132_1
y_{n-2}
134_3
b_3
132_2
y_{n-3}
132_3
134_{M-3}
b_{M-2}
132_{M-3}
y_{n-M+2}
134_{M-1}
b_{M-1}
132_{M-2}
y_{n-M+1}
134_M
b_M
132_{M-1}
y_{n-M}
FINITE IMPULSE RESPONSE (FIR)
INFINITE IMPULSE RESPONSE (IIR)
GENERAL DIGITAL FILTER ; ORDER MAX (N,M)

Fig.10A.

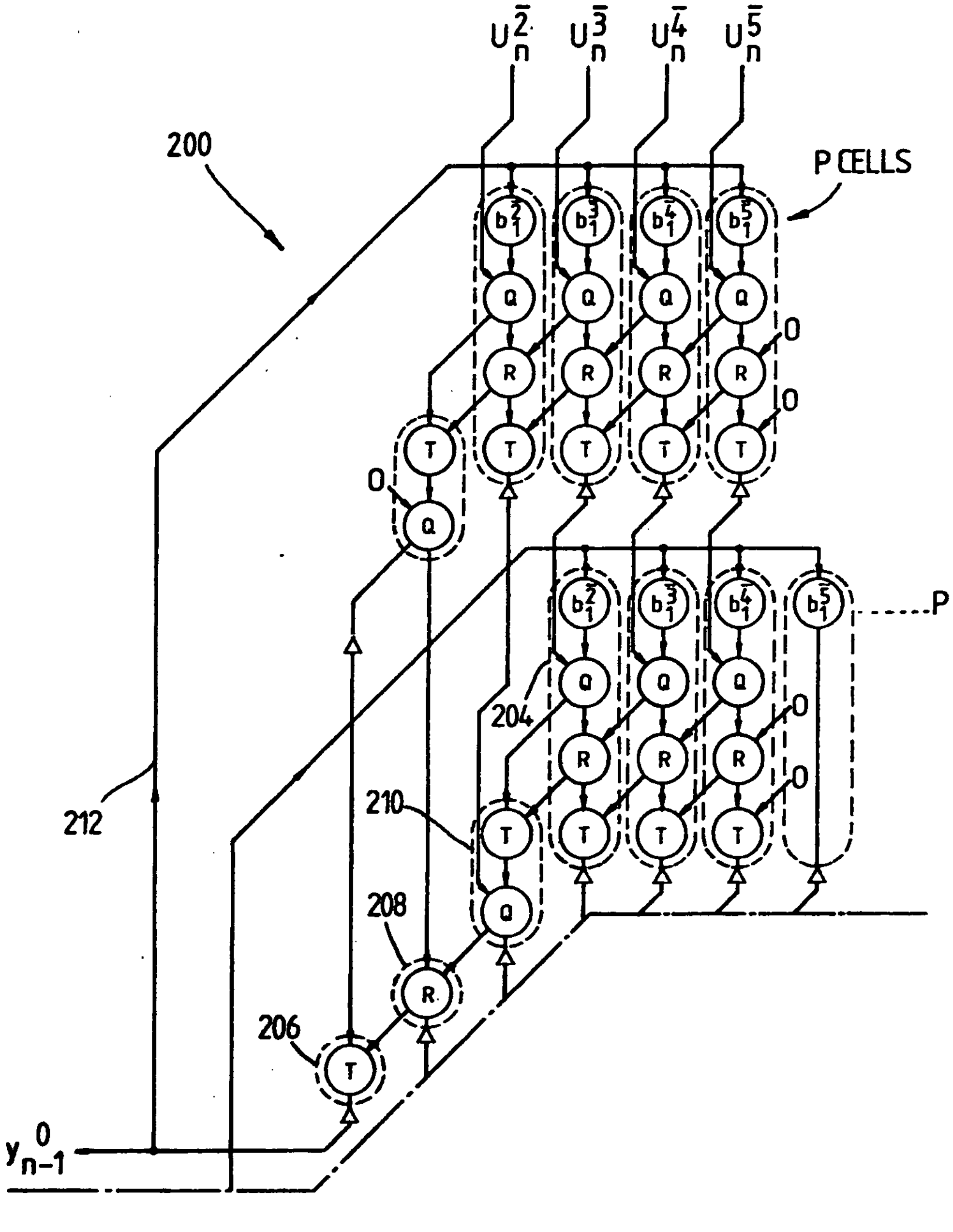
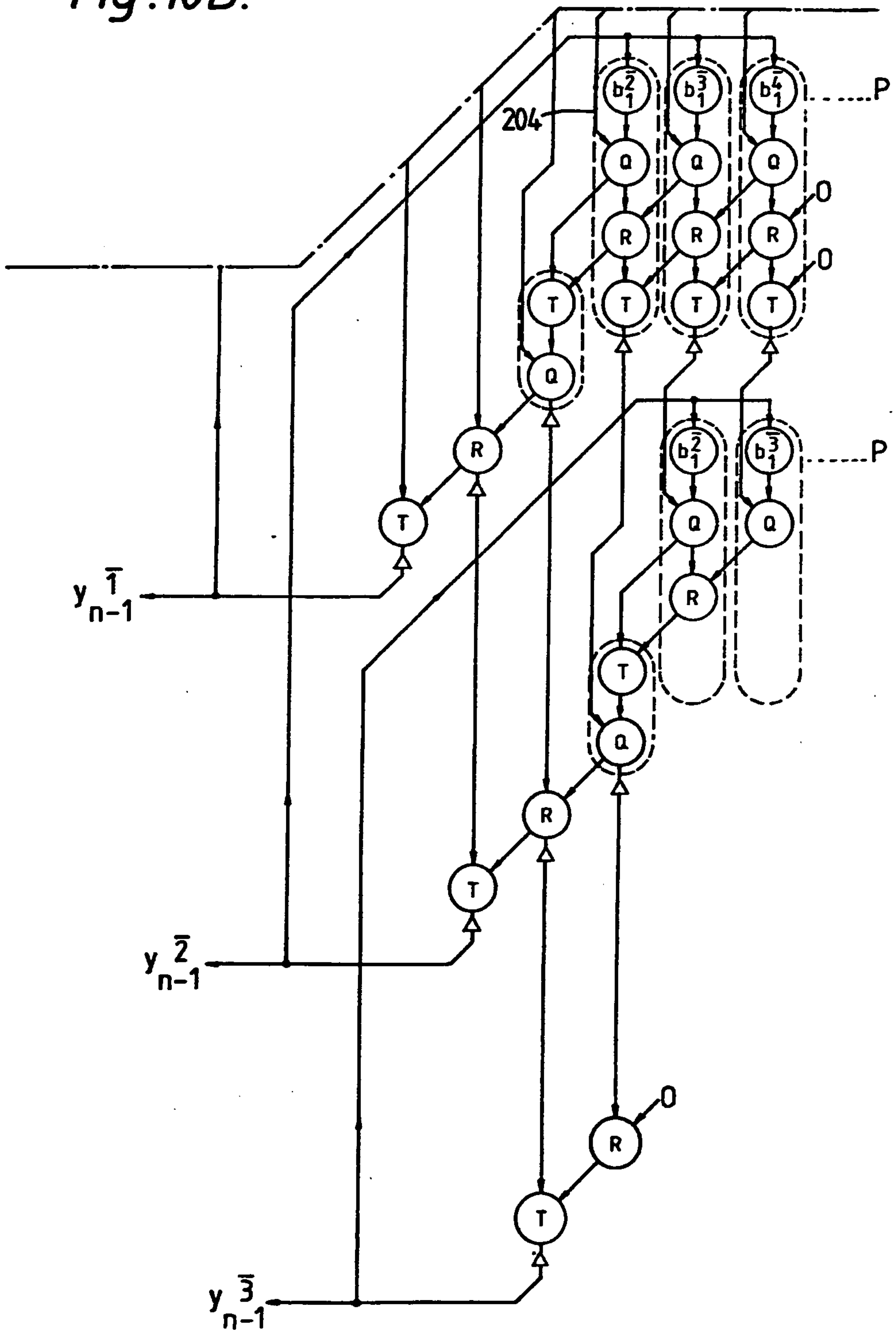

$U_n^{\bar{2}}$
$U_n^{\bar{3}}$
$U_n^{\bar{4}}$
$U_n^{\bar{5}}$
200
P CELLS
$b_1^{\bar{2}}$
$b_1^{\bar{3}}$
$b_1^{\bar{4}}$
$b_1^{\bar{5}}$
Q
R
T
0
P
204
210
208
206
212
y_{n-1}^0

Fig. 10B.



$b_1^{\bar{2}}$
$b_1^{\bar{3}}$
$b_1^{\bar{4}}$
P
204
Q
R
T
0
0
$b_1^{\bar{2}}$
$b_1^{\bar{3}}$
P
$y_{n-1}^{\bar{1}}$
$y_{n-1}^{\bar{2}}$
$y_{n-1}^{\bar{3}}$

Fig. 10C.

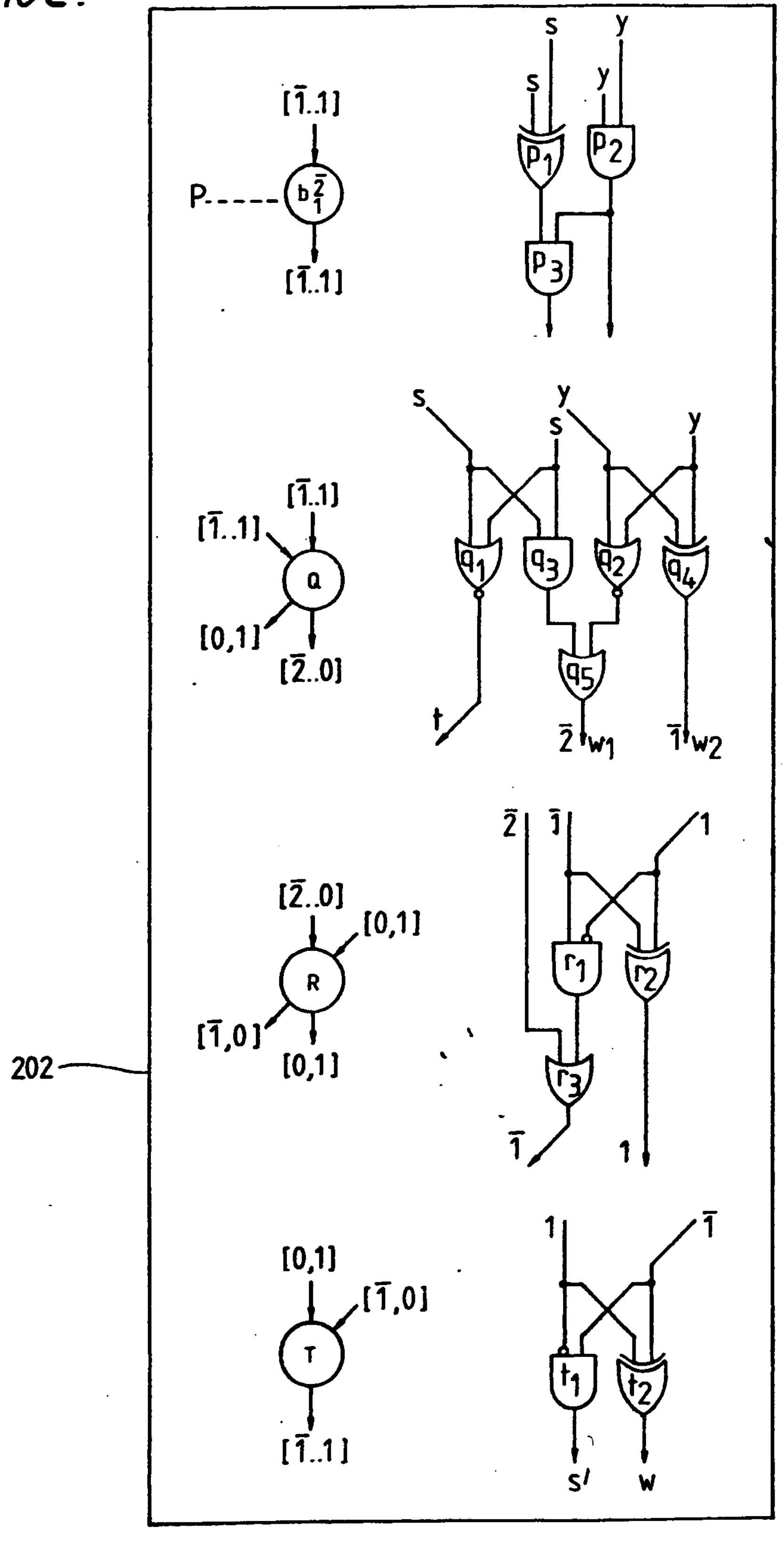

[1̄..1]
P
b 2̄ 1
[1̄..1]
s
s
y
y
p1
p2
p3
[1̄..1]
[1̄..1]
Q
[0,1]
[2̄..0]
s
s
y
y
q1
q3
q2
q4
q5
t
2̄ w1
1̄ w2
[2̄..0]
[0,1]
R
[1̄,0]
[0,1]
2̄
1̄
1
r1
r2
r3
1̄
1
[0,1]
[1̄,0]
T
[1̄..1]
1
1̄
t1
t2
s'
w
202

Fig.11A.

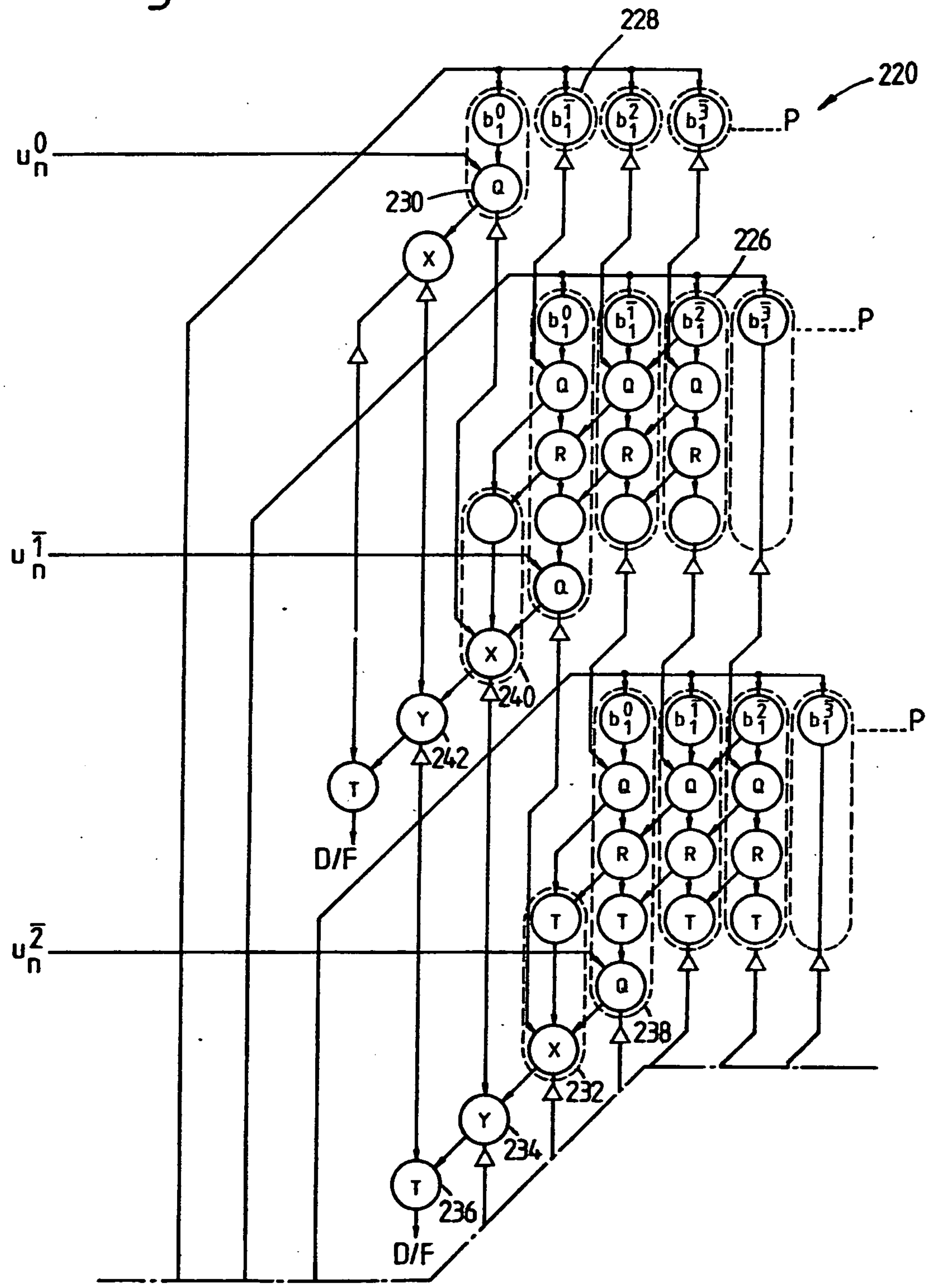

228
220
b_1^0
$b_1^{\bar{1}}$
$b_1^{\bar{2}}$
$b_1^{\bar{3}}$
P
u_n^0
Q
230
X
226
R
$u_n^{\bar{1}}$
240
Y
242
T
D/F
$u_n^{\bar{2}}$
238
232
234
236

Fig.11B.

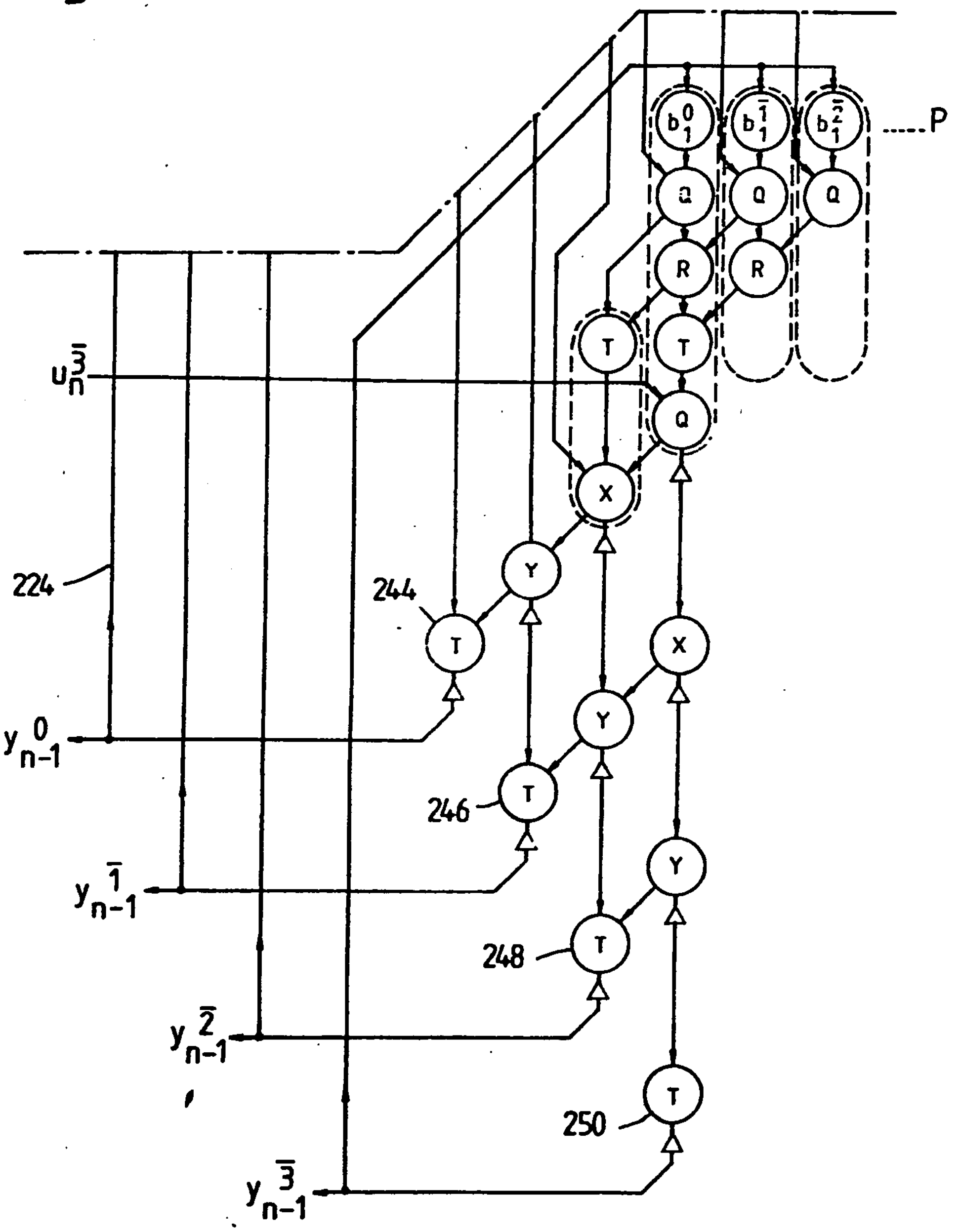
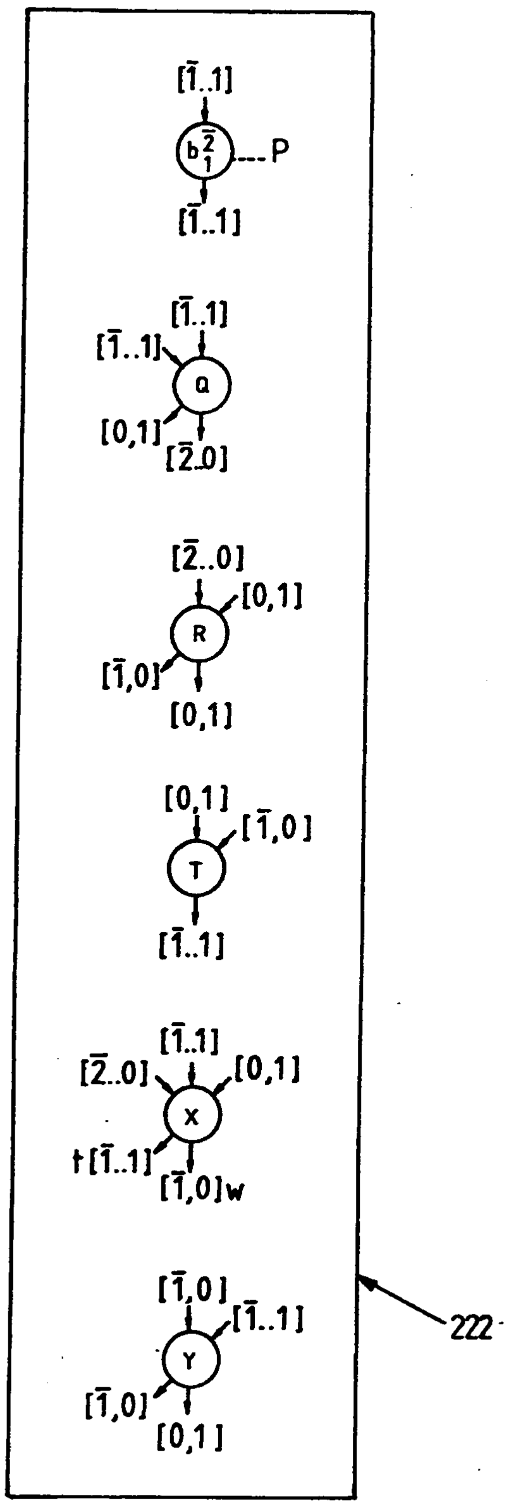

b_1^0
$b_1^{\bar{1}}$
$b_1^{\bar{2}}$
P
Q
R
T
X
Y
$u_n^{\bar{3}}$
224
244
246
248
250
y_{n-1}^0
$y_{n-1}^{\bar{1}}$
$y_{n-1}^{\bar{2}}$
$y_{n-1}^{\bar{3}}$

Fig. 11C.


[$\bar{1}$..1]
$b^{\bar{2}}_{1}$
P
[$\bar{1}$..1]
[$\bar{1}$..1]
[$\bar{1}$..1]
Q
[0,1]
[$\bar{2}$..0]
[$\bar{2}$..0]
[0,1]
R
[$\bar{1}$,0]
[0,1]
[0,1]
[$\bar{1}$,0]
T
[$\bar{1}$..1]
[$\bar{1}$..1]
[$\bar{2}$..0]
[0,1]
X
t[$\bar{1}$..1]
[$\bar{1}$,0]w
[$\bar{1}$,0]
[$\bar{1}$..1]
Y
[$\bar{1}$,0]
[0,1]
222

RECURSIVE PROCESSOR

This invention relates to a recursive processor, ie a processor arranged to recycle output results to its input for the production of further results.

Digital data processors for multiplication of numbers, vectors and matrices are available in the prior art, as are related devices for correlation and convolution. British Patent No 2,106,287B (Reference (1)) describes bit-level systolic arrays for (a) number-number multiplication, (b) matrix-vector multiplication and (c) convolution. British Patent No 2,144,245B (Reference (2)) describes a similar array for matrix-matrix multiplication, and British Patent No 2,147,721B (Reference (3)) relates to further developments for improvement of array efficiency. References (1) to (3) disclose arrays of logic cells with nearest neighbour row and column interconnections for bit propagation. Figure 1 of Reference (1) shows additional diagonal connections between second nearest neighbour cells. Each cell is a gated full adder with single bit inputs. It generates the product of two multiplicand bits, adds the product to input sum and carry bits and produces new sum and carry bits. The sum bits accumulate in cascade down array columns (or diagonals in Reference (1) Figure 1). Multiplicand bits propagate along array rows. One-bit intercell latches activated by clock signals provide for bit storage and advance between cells, and ensure that the arrays are pipelined at the cell or bit level. Where appropriate, the arrays include column output accumulators arranged to sum separately computed contributions to output terms.

Published British Patent Application Nos 2,168,509A, 2,187,579A and 2,192,474A (References (4), (5) and (6) respectively) demonstrate further bit-level systolic arrays which exhibit improved properties by the use of stationary multiplicative coefficients. Each coefficient remains associated with a respective cell, unlike References (1) to (3). However, data bits propagate along array rows for multiplication at gated full adder cells as before, and sum bits accumulate in cascade down array columns. Stationary array coefficients are also disclosed by Urquhart and Wood in the GEC Journal of Research, Vol 2, No 1, 1984, pp 52-55 (Reference (7)) and Proc IEE Part F, Vol 131 No 6, 1984, pp 623-31 (Reference (8)). These arrays also employ gated full adder cells with row and column interconnections.

One major area of application of bit-level systolic arrays is in the field of digital filters. Correlators and convolvers disclosed in References (1), (5) and (6) are examples of non-recursive, finite impulse response (FIR) filters. In digital signal processing, the correlation operation is defined by:

$$y_n = \sum_{i=0}^{N-1} a_i x_{n+i} \qquad (1)$$

where: a_i (i = 0 to N-1) represents a set of N correlation coefficients,
x_{n+i} is the (n+i)th input value,
y_n is the nth correlation result.

Successive values of x_{n+i} form an input data stream, and successive y_n values the filtered output stream.

Digital filters based on the prior art of References (1) to (8) are unsuitable for recursive filter applications, as the following analysis will show. These prior art arrays are pipelined at the bit level by clock-activated latches in the lines interconnecting neighbouring logic cells. This allows each array cell to compute a bit-level contribution to an output result while other cells are computing other contributions. Accordingly, data may be input on every clock cycle of operation without waiting for successive results to emerge from the array. Furthermore, the operating speed is not governed by the time taken for the whole array to compute a result. It is governed by the maximum clock rate of the latches associated with a single logic cell, which is much greater. However, against this, it is a basic feature of prior art systolic arrays that there is a time delay between data input and result output. In a typical case such as Reference (5), one row of logic cells is required per multiplicative coefficient in a coefficient set for convolution or correlation. In addition, an array output accumulator may be required to sum separately computed contributions to individual output terms. There is typically a delay of one clock cycle per row for arrays accumulating results down columns. To this must be added any output accumulator delay. In the case of a digital filter based on the Reference (5) bit-serial data input device, an N-point convolution or correlation with N p-bit coefficients provides a delay of N+2(p-1) clock cycles between input of a data bit and output of a result bit. Furthermore, for output results q bits in length, there is a delay of N+2(p-1) + (q-1) clock cycles between initiation of data input and output of the final bit of a result from the output accumulator. In the case of a 16-point convolution with 8-bit coefficients and data, which produces 20-bit results, the delay will be 49 clock cycles. If the array is clocked at 5 MHz, the delay is about 10 microseconds, and it is referred to as the "latency" of the processor. It does not give rise to difficulty in the case of FIR filters, since it merely means that there is an insignificant delay between initiation of data input and that of result output. Thereafter, input and output proceed at the same rate; ie input is received and output is generated on each clock cycle.

However, the latency of prior art digital processors gives rise to difficulty in the area of recursive processing, as required in infinite impulse response (IIR) filters. The simplest form of IIR filter is that where the output depends both on the input and on an earlier output. It is known as a "first-order section". The computation can be expressed in the form:

$$y_n = a_o x_n + a_1 x_{n-1} + b_1 y_{n-1} \qquad (2)$$

where x_{n-1} and x_n are successive data values in a continuously sampled stream, a_o, a_1 and b_1 are coefficients determining the filter response function, and y_{n-1} and y_n are successive output results.

Equation (2) may be rewritten:

$$y_n = u_n + b_1 y_{n-1} \qquad (3)$$

where

$$u_n = a_o x_n + a_1 x_{n-1} \qquad (4)$$

Equation (3) demonstrates that y_n is the sum of a non-recursive term u_n (depending only on input data) and a recursive term arising from the immediately preceding result y_{n-1}. This can be rewritten to express y_n in terms of y_{n-k} (k = 2, 3 ...) if required.

Any processor arranged to implement Equation (3) requires access to y_{n-1} (or an earlier result) in order to compute y_n. Accordingly the processor must compute and output y_{n-1} before beginning the computation of y_n. The characteristics of prior art processors now become much more serious, since their latency interval of many clock cycles must intervene between the computation of each pair of successive results. Instead of producing a new result every clock cycle, results are therefore spaced by the latency interval which may be 50 or more clock cycles. A latency of 50 cycles in a parallel recursive computation corresponds to the processor being only 2% efficient, or alternatively having an operating rate which is 1/50th that of a similar non-recursive processor.

The construction of digital filters has been discussed by R F Lyon in VLSI Signal Processing, a Bit Serial Approach, P B Denyer and D Renshaw, Addison Wesley, pp 253-262 1985. It is also described by Jackson et al in IEEE Trans on Audio and Electroacoustics, Vol AU-16, No 3 pp 421-423 1968. Neither of these addresses the problem of latency and inefficiency in IIR filters.

The latency problem is discussed by Parhi and Messerschmitt in ICASSP 87, pp 1855-1858. Their basic approach is to rearrange the algorithm expressed by Equations (3) and (4) above so that y_n becomes expressed in terms of y_{n-k}. They point out that the latency problem is inherent in recursive algorithms, but describe how it can be tolerated by a so-called "look-ahead" approach. In essence, this amounts to coping with latency by arranging the algorithm to employ as a recursive input whatever output is available. A parallel processor with a latency of k clock cycles will have y_{n-k} available at its output when u_n in Equation (3) is to be input. Since Equation (4) gives

$$y_n = a_o x_n + a_1 x_{n-1} + b_1 y_{n-1}$$

then

$$y_{n-1} = a_o x_{n-1} + a_1 x_{n-2} + b_1 y_{n-2} \tag{5}$$

and

$$y_{n-2} = a_o x_{n-2} + a_1 x_{n-3} + b_1 y_{n-3} \tag{6}$$

Expressing y_n in terms of y_{n-3}:

$$y_n = a_o x_n + a_1 x_{n-1} + b_1 \left[a_o x_{n-1} + a_1 x_{n-2} + b_1 \left\{ a_o x_{n-2} + a_1 x_{n-3} + b_1 y_{n-3} \right\} \right] \tag{7}$$

By induction, y_n in terms of y_{n-k} is given by.

$$y_n = \sum_{i=0}^{k-1} b_1^i \, [a_o x_{n-i} + a_1 x_{n-i-1}] + b_1^k \, y_{n-k} \tag{8}$$

The right hand side of Equation (8) consists of a non-recursive summation term together with a recursive term consisting of the product of y_{n-k} and a coefficient. Parhi and Messerschmitt have therefore dealt with the latency problem by choosing the feedback term y_{n-k} to be sufficiently early in the output y_n series for latency to be accommodated. However, the price they pay for this approach is the requirement to evaluate the Equation (8) summation term. This requires the addition of k terms, each of which involves a respective coefficient b_1^i multiplying the sum of two products of multibit numbers. This rapidly becomes unnamanageable as k increases, since each $a_o x_{n-i}$ alone would require a processing array as described in Reference (1). The Parhi et al approach consequently deals with latency, but only at the price of requiring an undesirably large non-recursive processor. For example, if k is 50 as for a typical prior art processor, the procedure requires the summation of fifty multiply twice, add, multiply operations.

It is an object of the present invention to provide an alternative form of recursive processor.

The present invention provides a recursive processor for multiplying successive recycled output results by a coefficient and adding their products to respective input terms, the processor including multiplier cells and accumulating means connected to form rows and columns, and wherein:

(1) each row is arranged to multiply a respective recycled digit by at least the more significant coefficients digits;
(2) the multiplier cells are associated with individual coefficient digits decreasing in significance along rows and increasing in significance down columns containing more than one such cell in each case;
(3) each multiplier cell is arranged to compute output sum and where necessary transfer digits corresponding to recycle¹ and coefficient digits multiplied together and where necessary added to input digits;
(4) each row begins with a respective accumulating means having at least such of the following functions as are required by availability of relevant neighbouring processor elements:

(i) receipt of its most significant cell's transfer digit output,
(ii) receipt of output sum digits from the respective preceding row's accumulating means and most significant multiplier cell,
(iii) processing received digits to generate output digits of differing significance, passing the most significant of such digits to a respective processor output and passing the remaining one or more digits to accumulating means in a respective succeeding row;

(5) the multiplier cells and accumulating means operate in accordance with signed digit number representation arithmetic involving digit redundancy and providing most significant digit first computation;
(6) the processor includes row neighbour interconnection lines for passage of transfer digits between adjacent multiplier cells in the direction of increasing coefficient digit significance and between each row's most significant multiplier cell and accumulating means;

(7) the processor includes column neighbour interconnection lines arranged for sum digit transfer down each column between accumulating means, between multiplier cells and between most significant multiplier cells and accumulating means where available in each case, the column interconnection lines including clock-activated storing means for sum digit storage and advance between rows; and

(8) each accumulating means has a respective most significant digit output connectable as a recycled multiplicand digit input to all multiplier cells of a respective row selected in accordance with accumulation arithmetic, recycled digit significance and coefficient magnitude.

The invention provides the advantage that it is suitable for performing recursive computations without prior art latency disadvantages. In one embodiment in particular, it is capable of performing the Equation (3) recursive computation (first order IIR filter section) with a latency of only two clock activation cycles irrespective of input word length or number of processor stages. Furthermore, unlike Parhi and Messerschmitt, it is necessary only to compute $u_n = a_0x_n + a_1x_{n-1}$ for input to the processor of the invention, instead of the unmanageably large non-recursive term given in Equation (8).

The invention achieves these advantages by being configured to operate in accordance with signed digit number representation arithmetic involving digit redundancy. In this form of arithmetic, each digit may take positive or negative values and there is more than one way of representing a number. A processor in accordance with the invention implementing this arithmetic computes output results most significant digit first and subsequent digits in descending order of significance. This means that the most important part of a result is available first for recycling to produce a successive result. In contradistinction, prior art devices compute least significant digits first, and must await propagation of carry digits up to more significant computations before producing a result for recycling.

A further consequence and advantage of the invention is that, in computations requiring accuracy only to digits of higher significance, circuitry devoted to lower significance computations may be omitted, unlike prior art devices. Each succeeding row of a processor configured in this way has progressively fewer processing elements, which leads to circuitry savings compared to the prior art.

Each first row multiplier cell may be arranged to add a respective digit of an input term to the product of its associated coefficient digit and multiplicand input. Alternatively, the most significant multiplier cell in each row may be arranged to add a respective input term digit to the product of its coefficient digit and multiplicand input. This latter arrangement provides for input to and output from the processor with like temporal skew, ie delay between adjacent digits. It facilitates connection of first and second processors of the invention in series to form a cascaded arrangement. Such an arrangement includes recycle connections from the first processor's accumulating means most significant digit outputs to multiplier cell multiplicand inputs of respective rows of both processors. Moreover, the second processor has accumulating means most significant digit outputs connected to respective most significant multiplier cell addition inputs of the second processor. It is suitable for use in computation of the second order IIR filter section relation.

In one embodiment, the invention is configured in accordance with radix 4 signed digit number representation arithmetic. It incorporates multiplier cells with multiply, full add and half add functions. Where required to accept temporally skewed input digits, it incorporates most significant multiplier cells with multiply and two full add functions. Processors which are incomplete because lower significance outputs are not computed may incorporate multiplier cells some of which have reduced addition functions. The multiply operation involves input digits in the maximally redundant radix 4 set (-3 to +3), coefficient digits and output sum digits in the range (-2 to +2) (minimally redundant set), and transfer digits in the range (-1 to +1). The accumulating means operation involves summation of transfer digits output from the most significant multiplier cell in a row with (where appropriate) the output sum digit from the most significant multiplier cell of the row above. This generates a sum digit for output to the row below, and a transfer digit for addition to the preceding row's accumulating means output sum digit to provide the row output or recycle digit.

In order that the invention might be more fully understood, embodiments thereof will now be described, by way of example only, with reference to the accompanying drawings, in which:

Figure 1	is a schematic drawing of a recursive processor of the invention;
Figure 2	illustrates input and output terms for cells in the Figure 1 processor;
Figure 3	schematically shows an alternative form of processor of the invention adapted for increased coefficient magnitude and temporally skewed input;
Figures 4,	5, 6 and 7 are simplified drawings of processors of the invention illustrating implementation as building blocks for various IIR filter functions;
Figure 8	is a more detailed drawing of the processor of Figure 6;
Figure 9	is a generalised representation of a digital filter;
Figure 10	is a further embodiment of a processor of the invention arranged to implement radix 2 arithmetic; and
Figure 11	is an alternative version of the Figure 10 embodiment arranged for skewed input and increased coefficient magnitude.

Referring to Figure 1, there is shown in schematic form a recursive processor 10 of the invention. The processor 10 incorporates processing cells in four rows 12_1 to 12_4 indicated by horizontal chain lines. The processing cells are of varying kinds to be referenced later. Successive rows 12_2 to 12_4 are right shifted compared to respective rows 12_1 to 12_3 immediately above, and this structure defines cells in seven columns 14_1 to 14_7 indicated by vertical chain lines. In the following description, cell location in the processor 10 will be identified by indices applied to corresponding reference numerals; ie X_{ij} would indicate an ith row, jth column cell of type X, where X is a reference numeral.

The processor 10 comprises three types of cell 16, 18 and 20, of which 16 is a composite such as first row, third column cell 16_{13} indicated within dotted lines. The cell 16_{13} incorporates three subcells, types A, B and C, which have differing functions. The subcell structure is shown to assist later analysis of mode of operation. Not all cells 16 incorporate all three subcells A, B and C.

The last one or two cells 16 on the right hand end of each row lack type C or type B and C subcells. The purpose of this representation is to illustrate an important advantage of the invention. Unlike the prior art it computes output digits in descending order of significance, the most significant digit appearing first as will be described later. Accordingly, when accuracy to a given number of digits is required, an array of the invention need only incorporate those cells and parts of cells which contribute to those digits. Other cells are omitted, which gives rise to the apparently incomplete processor structure illustrated.
All four rows 12_1 to 12_4 include respective type D cells 18_{11} and 18_{22} to 18_{44}. The second to fourth rows 12_2 to 12_4 include respective type E cells 20_{21} to 20_{43}.

Referring now also to Figure 2, the five cell and subcell types A to E are illustrated with labelled input and output digits. The type A subcell is a multiplier. It receives an input digit y_{in} from vertically above, and multiplies this by a single digit coefficient $b_i^{\overline{m}}$ (m = 1, 2, 3 or 4) with which it is labelled. It provides an intermediate sum output digit w_{out} vertically below, and a transfer digit t_{out} of relatively higher significance below and to the left. The type B subcell is equivalent in arithmetic terms to the type D cell; both are full adders, but they perform different roles in the processor 10. The expression "full adder" is used by analogy with binary logic to describe a three input device, a "half adder" being a two input device. Both types B and D receive three inputs from above, and add them to produce intermediate sum and transfer digits to be passed on below and below left respectively. However, type B has inputs y'_{in}, and w_{in} and t_{in} together with outputs w'_{out} and t'_{out}, whereas the equivalents for type D are y'_{in}, t_{in} and t'_{in} together with t''_{out} and w''_{out}. The digit nomenclature is intended to indicate the progress of a computation in the processor 10. Thus an unprimed digit arises prior to a primed digit, which in turn arises prior to a doubly primed digit. Moreover, a digit such as t'_{out} from a type B subcell is t'_{in} for a respective type D cell or type C subcell connected thereto. The nomenclature accordingly assists analysis to be given later.

The A subcells in each cell 16 receive multiplicand inputs recycled via feedback lines 30_1 to 30_4 connected to respective rows 12_1 to 12_4. This generates product and transfer digits w_{out} and t_{out}, of which w_{out} becomes a vertical input to the associated B subcell immediately below; t_{out} passes diagonally down to the neighbouring B subcell where available, or to a D cell otherwise. Each B subcell also receives a transfer digit output from a right hand neighbouring A subcell and an input from above. For first row cells 16, this last input is u_n^1 to u_n^4 on lines 32_1 to 32_4. For second, third and fourth row processing cells 16 containing B subcells, this input is received from the like cell in the same column of the row immediately above. Signals pass between rows (ie down columns) via clock-activated latches or storing devices indicated by triangles such as 34, all latches being clocked in synchronism.

Each B subcell adds its three inputs together to produce output sum and transfer digits w'_{out} and t'_{out}. The sum w'_{out} passes to the respective C subcell if available. The transfer digit t'_{out} passes diagonally down to a left hand neighbouring C subcell where available, or to a D cell otherwise. Each C subcell adds its two inputs from above and above right to produce a single digit output. This output passes via a latch 34 to a B subcell or a D cell in the same column and in the row immediately below. Like the equivalent E cells, each C subcell is a half adder without any transfer digit output.

Each D cell adds its three inputs from above, above upper right and above lower right (y'_{in}, t_{in}, t'_{in}) to produce intermediate sum and transfer digits w''_{out} and t''_{out}. The first row D cell has one input permanently zero, and for this cell t''_{out} must also be zero. It could in fact be replaced by a half adder type E cell. Each D cell provides w''_{out} as a w''_{in} input to the E cell of the row below where available, or to a lowermost array output 36_4 via two latches in series in the case of the last row D cell. Transfer digits t''_{out} from second to fourth row D cells pass as diagonal inputs to respective like row E cells for addition to w''_{out} from previous row D cells. Each E cell output is a single digit y_{out} which passes via a latch 34 to a respective array output 36_1 to 36_3. The array outputs 36_1 to 36_4 are connected to respective feedback lines 30_1 to 30_4.

The processor 10 operates on totally different principles as compared to prior art devices. They employ conventional binary arithmetic, in which successive digits (bits in binary arithmetic) are computed in ascending order of significance. In particular, least significant bits are computed first and carry bits propagate to computations of higher significance. In consequence, the greater the significance or importance of a digit, the longer it takes to emerge in prior art devices. Furthermore, the value of the least significant digit affects those of higher significance; eg adding 0001 to 0111 produces 1000 in binary arithmetic. If the least significant bit is not computed to reduce processor size and processing time, the most significant bit is wrong in this example. The processor of the invention operates in accordance with so-called redundant number arithmetic based on signed digit number representations (SDNRs). These differ from conventional arithmetic in that any digit may have a positive or a negative value, and also in that there is more that one way of representing a number. Furthermore, there is normally a choice of digit sets. Before discussing the way in which the processor 10 executes a computation, this form of arithmetic will be discussed.
In conventional radix 10 (decimal) arithmetic, the digit set is 0 to 9 and the digits are all of the same sign. The processor 10 is constructed to operate with an SDNR of radix 4 and digit sets such as -3, -2, -1, 0, 1, 2, 3. This is known as the maximally redundant set. It is also possible to employ other digit sets with this radix such as the minimally redundant set -2 to 2. The redundancy aspect is exemplified by the decimal number 15, which in the maximally redundant radix 4 digit set is (1,0,-1), (0,3,3) or (1,-3,0,-1) etc. Generally speaking, for a radix r the maximally and minimally redundant sets have (2r-1) and (r+1) digits respectively. For radix 2 these are the same, a set of three digits 1, 0 and -1.

The processor 10 operates on fractional quantities, so that recursion or repeated feedback from ouptut to input does not produce overflow. Thus inputs $\overline{u}_n^1$ to $\overline{u}_n^4$ (the digits of the nth value of u) at 32_1 to 32_4 have digit significance of 4^{-1} to 4^{-4} respectively, ie 1/4 to 1/256. Similarly, the (n-1)th result output digits y_{n-1}^0 to $\overline{y}_{n-1}^3$ at 36_1 to 36_4 have significance 4^0 to 4^{-3} respectively, or 1 to 1/64. In addition, the digits $\overline{b}_1^1$ to $\overline{b}_1^4$ of the multiplying coefficient b_1 have the same significance as $\overline{u}_n^1$ to $\overline{u}_n^4$. In consequence, each column 14_1 to 14_7 of the processor 10 is of constant digit significance, and this significance reduces by one per column from left to right. In cell 16_{13} for example, subcell A multiplies an input y_{n-1}^{0} by $b_1^{\overline{2}}$ to produce a lower significance digit w_{out} to be passed on in the same column, together with a higher significance digit t_{out} to be passed to the left. The digits w_{out} and t_{out} have significances 2 and 1 respectively. The columns 14_1 to 14_7 therefore correspond to digit significances 0 to 6 respectively. The processor 10 is designed for values of b_1 in the range −0.2222 to +0.2222 in radix 4, equivalent approximately to −0.66 to +0.66 in decimal.

Individual digits of the kinds y and u may take any value in the maximally redundant radix 4 set (−3 to 3). However, digits of the kinds b_1 and w are restricted in this example of the invention to the minimally redundant radix 4 set (−2 to 2). Moreover, transfer digits of the kind t are restricted to values (−1 to 1). These digit value restrictions apply irrespective of whether or not the relevant digit is primed or carries a subscript.

Having clarified the arithmetic rules implemented by the processor 10, its operation will now be discussed in more detail. Figure 1 illustrates the first row 12_1 of the processor 10 receiving input of the four digits $u_n^{\overline{1}}$ to $u_n^{\overline{4}}$ representing the nth non-recursive input u_n. Each row 12_m (m = 1 to 4) is also receiving a respective feedback digit $y_{n-1}^{\overline{m-1}}$ generated in the preceding or (n−1)th computation. The processor 10 is designed to implement the first order IIR filter section relation given in Equation (3) and repeated below for convenience:

$$y_n = u_n + b_1 y_{n-1} \qquad (3)$$

The first row 12_1 of the processor 10 implements multiplication of the most significant digit y_{n-1}^{0} of the (n−1)th output result y_{n-1} by the coefficient b_1 having digits $b_1^{\overline{1}}$ to $b_1^{\overline{4}}$. Moreover, the first row 12_1 provides for addition of the non-recursive term u_n with digits $u_n^{\overline{1}}$ to $u_n^{\overline{4}}$. The first step in this row is for each type A sub-cell to receive input of y_{n-1}^{0} and multiply it by the respective digit $b_1^{\overline{m}}$ (m = 1 to 4). Processing cell 16_{13} within dotted lines will be considered first. Its multiplication operation involves a digit y_{n-1}^{0} in the range (−3 to 3) and a digit $b_1^{\overline{2}}$ in the range (−2 to 2). The product is therefore in the range (−6 to 6). However, the allowed range for w_{out} from a type A subcell is (–2 to 2), so a transfer digit t_{out} is generated which is one level higher in significance. This is expressed by:

$$w_{out} + rt_{out} = b_1^{\overline{m}} \cdot y_{in} \qquad (9)$$

where r is the radix and m is the multiplier digit significance; r = 4, and m = 2 in the present case.

Since r = 4 and t_{out} = ± 1 or 0, rt_{out} = ± 4 or 0. Values of the product $b_1^{\overline{m}} \cdot y_{in}$ from –2 to 2 are expressed by $t_{out} = 0$ and $w_{out} = b_1^{\overline{m}} \cdot y_{in}$. Values ± 3 of the product are expressed by $w_{out} = \mp 1$ and $t_{out} = \pm 1$. Values ± 4 to ± 6 are expressed by w_{out} = ± 0 to ± 2 and $t_{out} = \pm 1$. All possible products of the A subcell multiplication are therefore accommodated within the digit limits previously given.

The A subcell of cell 16_{13} passes the transfer digit t_{out} diagonally down to the left hand neighbouring B subcell of cell 16_{12}, where it becomes t_{in}. It is added to $u_n^{\overline{1}}$ and to w_{out} of the A subcell of cell 16_{12}, with which it shares like digit significance. These three digits have value ranges (–1 to 1), (–3 to 3) and (–2 to 2) respectively. Their sum consequently lies in the range (–6 to 6), which is the same as the output range from a type A multiplier subcell. It can accordingly be accommodated by similar output digits w'_{out} and t'_{out} in ranges (–2 to 2) and (–1 to 1) respectively. The B subcell function is expressed by

$$w'_{out} + rt'_{out} = y'_{in} + t_{in} + w_{in} \qquad (10)$$

(y'_{in} is replaced by u_{in} for top row B subcells.)

The sum digit w'_{out} from the B subcell passes to the associated C subcell where available, which has the function

$$y'_{out} = w'_{in} + t'_{in}$$

Of these, t'_{in} arises from a neighbouring B subcell, and y'_{out} passes as input to the row below.

The transfer digit t'_{out} of the B subcell passes to the adjoining column's D cell 18_{11}, where it is added to the multiplier or A subcell transfer digit t_{out}. Other D cells 18_{22} to 18_{44} receive input of y'_{out} (range -3 to 3) from a C subcell in the row above. Cell 18_{11} has no row above however, and its y'_{in} input is set permanently to zero. Moreover, since it is adding two transfer digits both in the range -1 to 1, its output is in the range -2 to 2. Consequently, its transfer digit output t''_{out} is therefore permanently zero, and does not require to be connected. Hence cell 18_{11} has a y'_{in} input and a t''_{out} output which are unconnected. More generally, for other D cells 18_{22} to 18_{44}, the function is given by

$$w''_{out} + rt''_{out} = y'_{in} + t_{in} + t'_{in} \qquad (11)$$

Equation (11) is similar to Equation (10); y'_{in} is in the range (-3 to 3), and the equivalent for t_{in} and t'_{in} is (-1 to 1). The combined range is (-5 to 5), which can be accommodated by $r = 4$, t''_{out} in the range (-1 to 1) and w''_{out} in the range (-2 to 2) as previously described regarding B subcell.

The D cell 18_{11} provides the sum w''_{out} of both first row, second column transfer digits t_{in} and t'_{in} as an input w''_{in} to the second row E cell 20_{21}. This E cell also receives input of a transfer digit t''_{in} generated as t''_{out} from the second row D cell 18_{22}. The inputs to E cell 20_{21} are both of digit significance 0, since both are one level higher than that of column 14_2; ie E cell 20_{21} is summing w''_{out} (which arose from two first row, second column transfer bits) with a second row, second column transfer bit t''_{out}. Since w''_{out} and t''_{out} are in the ranges (-2 to 2) and (-1 to 1) respectively, E cell 20_{21} produces their sum y_{out} in the range (-3 to 3). This involves no digit significance increase, since the range is acceptable for terms of type y. The sum y_{out} was computed inter alia from most significant digits y^0_{n-1} and $u^{\overline{1}}_n$, and forms the most significant digit y^0_n of the result y_n succeeding that illustrated in Figure 1. The computation producing y^0_n passes through the two left hand first row latches 34 in parallel, and subsequently through the left hand second row latch 34 under E cell 20_{21}; y_n^o accordingly takes two clock cycles to emerge from the processor 10 after the preceding y_{n-1}^o is generated.

The digit $y_n^{\bar{1}}$ of second highest significance of the succeeding result y_n is formed similarly. The equivalent preceding digit $y_{n-1}^{\bar{1}}$ is fed back from the third row output 36_2 via the line 30_2 to each of the second row A subcells of cells 16_{23} to 16_{26}. It is formed one clock cycle later than y_{n-1}^o, and is one level lower in digit significance. Accordingly, its multiplication by $b_1^{\bar{1}}$ to $b_1^{\bar{4}}$ will yield products one level lower than those involving like multiplying coefficients in row 12, above. This is accommodated as shown in Figure 1 by the shift of row 12_2 by one column to the right relative to row 12_1.

The computation of $y_n^{\bar{1}}$ is similar to that of y_n^o, and so it will be described in outline only. It is produced as y_{out} from third row E cell 20_{32}, and is the sum of the transfer digit t''_{out} from the third row D cell 18_{33} and the digit w''_{out} from the second row D cell 18_{22}. Of these, cell 18_{22} sums the lower order digit y'_{out} from its column neighbour 16_{12} in the row above with transfer digits from its row neighbour 16_{23}. In turn, cell 16_{23} receives two transfer digits from its neighbour 16_{24}. The other input to third row E cell 20_{32} from its row neighbour D cell 18_{33} arises from the C subcell of cell 16_{23} in the row above and transfer digits from like row member 16_{34}. In turn, cell 16_{23} receives input from its column neighbour C subcell in the row above.

The digit $y_n^{\bar{2}}$ third in order of significance is formed similarly to $y_n^{\bar{1}}$, except that rows and columns further down and to the right become involved in computation. It is output from fourth row E cell 20_{43}. As has been discussed, its value can be affected only by outputs from cells up to three steps to the right in row 12_4 and up to four steps to the right in row 12_3. Of these, subcells B and C (not shown) of cells 16_{37} and 18_{46} and subcells C (not shown) of cells 16_{36} and 18_{45} cannot contribute to $y_n^{\bar{2}}$ and are omitted from the processor 10. This omission does not affect the value of $y_n^{\bar{2}}$, or indeed those of y_n^o and $y_n^{\bar{1}}$, which are correctly calculated. The processor 10 therefore contains the minimum of circuitry necessary to calculate the three highest significance result digits.

The least significant result digit $y_n^{\overline{3}}$ is derived as w''_{out} from fourth row D cell 18_{44}. It passes via two latches 34 to the lowermost processor output 36_4. It may be inaccurate by 1, ie if its calculated value is x in the range (−2 to 2), its actual value will be x−1, x or x+1. This corresponds to a truncation error in the final result. The lowermost latch 34 in Figure 1 is the only remaining item of a fifth row of cells, and corresponds to a latch following a type E cell output. It acts as a single digit accumulating means, there being no row neighbours generating other digits.

The foregoing discussion of mode of operation has been restricted to details of computation. Timing of processor operation will now be described. Initially, the processor outputs 36_1 to 36_4 are zero. The digits $u_1^{\overline{1}}$ to $u_1^{\overline{4}}$ of u_1, the first non-recursive input term, are fed to the processor's first row B subcells. Time is subsequently allowed for the first row cells to settle to final states. The processor latches (eg 34) are then clocked in synchronism by a common system clock (not shown) connected thereto. This advances the first row cell outputs to the second row for a second cycle of computation. After the settling time interval, the latches 34 are clocked once more. This advances the most significant digit y_1^0 of the first result y_1 from E cell 20_{21} to the uppermost processor output 36_1, where it is fed back to first row cells. In addition, intermediate results from other second row cells pass via latches 34 to third row cells. At the same time, ie after two clock cycles, the digits $u_2^{\overline{1}}$ to $u_2^{\overline{4}}$ of the second non-recursive term u_2 are input to first row cells. The processor latches are clocked a third time to advance first row outputs to second row cells. This also brings the second most significant digit $y_1^{\overline{1}}$ to the second row output 36_2, which is one cycle later than the appearance of y_1^0 at 36_1. On its appearance at 36_2, $y_1^{\overline{1}}$ is fed back as an input to second row cells.

The latches 34 continue to be clocked in synchronism, with successive values of u_n being input to the first processor row 12_1 every two clock cycles. Similarly, successive values of output digits $y_n^{\overline{m}}$ emerge (m+2) clock cycles later, where m = 0 to 3; ie there is a one clock cycle delay between emergence of digits differing in significance by one. In general, the digit $y_n^{\overline{m}}$ of mth significance of the nth result y_n appears at output 36_{m+1} after (2n+m) clock cycles, and is fed back for input to processor row 12_{m+1}. This illustrates an important feature of the processor 10, that each output digit $y_n^{\overline{m}}$ is recycled for subsequent use as soon as it is generated. Furthermore, as has been said, there is a delay of two clock cycles between input of u_n and output of y_n^o, but this is independent of input or output word length (number of digits per term u_n or y_n). Successive values of each of y_n^o to $y_n^{\overline{3}}$ appear every two clock cycles, so the processor 10 is 50% efficient. The efficiency may be improved to 100% when two independent operations are to be executed, since they may be interleaved and computed on alternate cycles. The input non-recursive term would be u_n on even clock cycles and u_n^* say on odd clock cycles. These efficiency values of 50% or 100% compare very favourably with much lower prior art values involving digital processors calculating least significant digits (bits) first. Efficiency may also be improved by combining the functions of two neighbouring rows into that of a single row. Alternatively, y_n may be expressed in terms of y_{n-2} as described with reference to Equations (4) to (7). The advantages of the invention arise from processor design based on signed digit number representation arithmetic with digit redundancy. This allows most significant digits to be computed first, and makes transfer digit propagation of short finite length.

The foregoing analysis demonstrates that the processor 10 executes the recursive first order IIR filter section computation given by Equation (3) as

$$y_n = u_n + b_1 y_{n-1} \qquad (3)$$

provided that u_n is computed separately, where:

$$u_n = a_o x_n + a_1 x_{n-1} \qquad (4)$$

Here x_n represents a continuous data stream and a_o, a_1 and b_1 are coefficients determining the filter frequency response; u_n may be calculated repeatedly separately from the processor 10, and employing the data stream x_n as input to simple prior art multipliers described for example in British Patent No 2,106,287B. It is non-recursive, so that feedback or recursion loop delay problems do not arise. Moreover, it involves only two multiplication operations followed by addition.

The processor 10 incorporates latches 34 between adjacent rows 12 but not between adjacent columns 14. It is in fact possible to introduce equivalent latches between columns, provided that inputs to first row cells have relative delays increasing with decreasing digit significance. However analysis of this configuration shows that the effect is to slow down processor operation. Prior art devices normally employ latches between both row and column neighbour cells.

The processor 10 accepts successive input terms u_n with digits $u_n^{\overline{m}}$ (m = 1 to 4) supplied in synchronism to first row multiplier cells 16_{12} to 16_{15}. It accepts an input and produces an output every two latch or clock activation cycles. It produces output terms y_n with digits $y_n^{\overline{m}}$ (m = 0 to 3) (2+m) clock cycles after input of u_n; ie the most significant digit y_n^o arrives after two clock cycles and successive digits are increasingly more delayed. This is referred to in the art of array processors as a "temporal skew". The processor 10 is therefore characterised by synchronous input but temporally skewed output. It is more convenient in some circumstances to provide a processor characterised by input and output of like temporal skew to facilitate connection in cascade. Moreover, the processor 10 is restricted as has been said to values of coefficient digits $|b_1^{\overline{1}}|$ etc not greater than 2. The maximum value of the coefficient b_1 is therefore 0.2222 in radix 4, or about 0.66 when converted to radix 10. It may be convenient to employ larger values of b_1 to provide more freedom of choice of filter response characteristics. The penalty for this is that input data values may then produce overflow.

Referring now to Figure 3, there is shown a further processor 50 of the invention. It is a modified version of that shown in Figure 1 to implement skewed input of u_n terms and multiplication by coefficients in the larger range of about -2.6 to +2.6 in radix 10. Cell inputs and outputs are illustrated in a box 51. The processor 50 is of very similar construction to the processor 10, and consequently its description will be largely restricted to aspects of difference. It multiplies by coefficient digits b_1^o to $b_1^{\overline{3}}$ contained in respective multiplier cells of each row. Each digit can take any values in the range (-2 to +2), and so the value of b_1 is in the range -2.222 to +2.222 in radix 4.

The processor 50 contains multiplier cells 52 and type D' and E accumulator cells 54 and 56 arranged in rows 58_1 to 58_4 and columns 60_1 to 60_7. Latches indicated by triangles such as 62 provide pipelining as before between column neighbours but not row neighbours. Ignoring subcells which are absent since not required for higher output digit significance, most of the multiplier cells 52 have the same structure as cells 16. They incorporate subcells A, B, B' and C as shown at 51 providing multiply, full add and half add functions. However, the most significant multiplier cell $52_{j,j+1}$ (j = 1 to 4) in each row 54_j incorporates an A subcell together with successive B and B' subcells. The C subcell of other multiplier cells is therefore replaced by a type B' subcell, and this provides for an additional addition function via input lines 63_1 to 63_4. The B' subcells have an identical function to that of B subcells, but input and output signals differ as shown in the box 51. The lines 63_1 to 63_4 supply respective input digits u_n^0 to $u_n^{\bar{3}}$ to most significant multiplier cells 52_{12} etc. However, synchronous input digits $v_n^{\bar{1}}$ to $v_n^{\bar{4}}$ may also be supplied to top row B sub-cells as illustrated, in a similar manner to inputs $u_n^{\bar{1}}$ etc in the Figure 1 processor 10.

The effect of exchanging a C subcell for a B' subcell is to introduce an extra transfer digit t'_{out} from the most significant multiplier cells 52_{12} etc. These cells now produce three transfer digits, as opposed to two for other multiplier cells such as 52_{13}. The type D' accumulator cells in at least the second row onwards must therefore accept four inputs, one sum digit of type y' (range −3 to +3) and three transfer digits (range −1 to +1). The range of the sum of these digits is (−6 to +6), which can be accommodated by output sum and transfer digits of types w and t. As previously described with reference to Equation (11), D cells in the processor 10 had inputs with a sum in the range (−5 to +5), whereas the output expression w + rt has a range (−6 to +6). The full output range of the D cell was therefore not exercised, and it can accommodate an extra transfer digit input as provided in the processor 50. The prime superscript to D' cells accordingly indicates a similar function to that of D cells, but including an additional transfer digit input.

In consequence of the increase in maximum size of the coefficient $|b_1|$ to 2.66, recycled output digits $y^{\overline{p}}$ generate products of higher digit significance as compared to the equivalent for the processor 10. To compensate for this, output digits are fed back to the next but one preceding row (as opposed to the preceding row in the processor 10). Each output digit $y^{\overline{p}}_{n-1}$ (p = 0 to 3) is recycled via a line 64_{p+1} to the respective row 58_{p+1}. The most significant output digit y^{o}_{n-1} will be multiplied inter alia by most significant coefficient digit b^{o}_{1}. This produces a product with two digits, t_{out} of significance +1 and w_{out} of significance 0. The latter is of the same significance as y^{o}_{n-1}. It must therefore be produced in the same processor column, since as previously indicated each column is associated with sum and transfer digits of constant significance. The selection of the row to which each output digit is recycled is in accordance with the significance of the recycled digit in combination with that of the most significant coefficient digit, the latter indicating coefficient magnitude. To illustrate this, consider the recycling of $y^{\overline{1}}_{n-1}$. When multiplied by b^{o}_{1}, this digit will provide sum and transfer digits of significance $\overline{1}$ and 0 respectively. The transfer digit will be added to other digits to produce an output digit of significance 1; ie the multiply and add operations introduces a digit significance increase of 2, as compared to that of the sum digit arising from multiplication and equal to the digit significances of $y^{\overline{1}}_{n-1}$ and b^{o}_{1} added together. Each recycled digit therefore generates a contribution to another digit two levels higher in significance in consequence firstly of multiplication by b^{o}_{1} and secondly of accumulation with other digits in accordance with radix 4 arithmetic. Each recycled digit must accordingly be fed back to the next but one preceding row. In comparison, the processor 10 operates with like arithmetic, but its maximum coefficient digit $b^{\overline{1}}_{1}$ is one level lower in digit significance. Each of its recycle digits can only produce a contribution to a respective output digit one level higher in significance. It therefore requires feedback of each recycle digit only to the respective preceding row. If the most significant coefficient were to be of significance 2, 3 or smaller, feedback or recycling would be in the same row or a subsequent row to compensate for decreasing coefficient magnitude. Moreover, other processors of the invention using radices other than 4 may produce a recycle digit significance increase of more than 2 in consequence of accumulation. The selection of the row to receive feedback is adjusted to compensate for this.

Generally speaking, recycle row selection is in accordance with accumulation arithmetic, coefficient magnitude and recycle digit significance.

Because the processor 50 recycles between non-adjacent rows, contributions to output results must pass through three latches 62. A new result is therefore generated every three cycles after input, and the generation and input rates are once every three cycles. This is slower than the processor 10, and is a consequence of increasing coefficient size.

The processor 50 also differs from the processor 10 in that it has two overflow outputs O/F. These are the transfer digit output of the first row D' cell 54_{11} and the sum output of the second row E cell 56_{21}. The need for this arises as follows. The coefficient b_1 may take values in the range +2.66 to -2.66 (approximately) in radix 10, and each u_n has a maximum value approaching 4.

Equation (3) provides:

$$y_n = u_n + b_1 y_{n-1}$$

Operation of Equation (3) allows y_n to increase without limit as n increases if b_1 and u_n are large enough. This is a basic characteristic of any recursive processor such as an IIR filter. The net effect of increasing allowed maximum values of b_1 from $|0.66|$ (processor 10) to $|2.66|$ (processor 50) is to produce the possibility of overflow. To avoid overflow, either the input non-recursive terms u_n (and/or v_n) or the coefficient b_1 must be restricted in value. A non-zero value at either of the outputs at O/F indicates that this restriction has not been observed, and provides an alarm.

The output digits $y_{n-1}^{\bar{2}}$ and $y_{n-1}^{\bar{3}}$ are output from cells 54_{44} and 52_{45} via two and three latches 62 respectively. These latches preserve the temporal output skew, ie the one cycle delay between output digits of adjacent significance. They also correspond to vestigial extra rows lacking multiply/add functions as previously described.

The processor 50 contains a respective B subcell in each first row multiplier cell for the purposes of adding a non-skewed input v_n. If this is not required, the B subcells may be omitted. First row cells would then contain only A and C subcells, apart from the most significant first row cell which would retain a B' subcell instead of a C subcell to add u_n^0.

Referring now to Figures 4 to 7, in which like parts are like-referenced, there are shown schematic functional drawings illustrating use of the processor 50 as a building block for use in IIR filter applications. For illustrational clarity, individual digits of terms being processed are not shown, and inputs of the kind v_n are not employed.

The processor 50 is illustrated in open loop form as a building block allowing any required connection scheme, ie there is (as yet) no recycle connection. It receives an input term u with temporally skewed digits indicated by a diagonal line 70. It receives a second skewed input y' (diagonal line 72), and y' is multiplied by a coefficient b at 74. The product by' is added to u at 76, and the sum passes via a system delay represented by clock-activated storing means 78 to provide an output y at 80. The output signal y has a temporal skew indicated by a line 81 and equivalent to those of inputs u and y'.

In Figure 5, the processor 50 is shown arranged as a first order section IIR filter. This merely requires a connection 82 from the output 80 to provide a multiplicand input at 74. By virtue of the delay introduced at 78, u_{n+1} provides an input 70 simultaneously with output of y_n at 80, y_n being given by:

$$y_n = u_n + b_1 y_{n-1}$$

which is Equation (3) as before. On the subsequent processing cycle (three clock cycles later), ie one processing cycle after that illustrated in Figure 5, u_{n+2} will provide the input 70, and the output at 80 will be y_{n+1} given by

$$y_{n+1} = u_{n+1} + b_1 y_n \qquad (12)$$

Accordingly, by induction, a stream of input terms u_n (n = 0,1,2,...) gives rise to a stream of output terms y_n representing the input filtered by a first order IIR filter section.

Figure 6 schematically illustrates two processors 50 and 50' connected in cascade to form a second order filter section. They are each equivalent to that shown in Figure 4, and like parts are like-referenced. However, parts of the upper processor 50' have prime indices to distinguish them from the accompanying processor 50, and they are associated with respective multiplier coefficients b_2 and b_1.

The output 80 of the processor 50 is connected to a recycle line 82 providing multiplicand inputs at 74 and 74' to both processors 50 and 50' simultaneously. The output 80 is illustrated at a time when it has received y_n from storing means 78. At this time, the upper processor 50' is receiving an input 70' of u_{n+2}, which consequently becomes added by adder 76' to b_2y_n from multiplier 74'. The output of adder 76' input to storing means 78' is therefore $u_{n+2} + b_2y_n$. Simultaneously, the output of storing mean 78' is the earlier equivalent of this involving the input term u_{n+1}, ie $u_{n+1} + b_2y_{n-1}$. The output of storing means 78' is the input to adder 76, where it is added to b_1y_n generated by multiplier 74 from y_n recycled via line 82. This produces $y_{n+1} = u_{n+1} + b_1y_n + b_2y_{n-1}$ at the input to storing means 78 simultaneously with output therefrom of y_n. The expression for y_n is given by replacing n+1 by n in that for y_{n+1}, which provides

$$y_n = u_n + b_1y_{n-1} + b_2y_{n-2} \qquad (13)$$

Equation (13) is the general expression for the recursive portion of a second order IIR filter section, which demonstrates that two cascaded processors of the invention can provide such a filter. As illustrated in Figure 7, three processors 50, 50' and 50" of the invention may be cascaded to provide a third order IIR filter section, and additional processors may be added to construct higher order filters.

Referring now to Figure 8, the second order IIR filter section of Figure 6 is shown in more detail. Cell reference numerals have been omitted to reduce complexity. It comprises two processors equivalent to 50 and 50' and referenced accordingly, but these are modified as previously described by omission of unnecessary top row B sub-cells, there being no v_n input. In consequence, the top row D' cell of each processor 50/50' cannot produce an overflow output. This can only occur at a second row E cell. Successive digits $u_n^{\overline{m}}$ are input to the most significant multiplier B' sub-cell of the left hand processor 50'. Intermediate output digits are fed from the processor 50' via lines 90_1 to 90_4 to respective most significant multiplier B' sub-cells of the processor 50. Recycle or feedback digits y_{n-1}^{0} to $y_{n-1}^{\overline{3}}$ are fed from the processor 50 as multiplicand inputs to respective rows of <u>both</u> processors 50 and 50' via lines 92_1 to 92_4.

Referring now to Figure 9, there is shown a schematic representation of a general digital filter 120. It has a left hand or input portion 122 comprising a chain of latches 124_0 to 124_{N-1} connected in series. An input x_n is connected to multiplier 126_0 and latch 124_0. The outputs of the latches are connected to respective multipliers 126_1 to 126_N associated with multiplicative coefficients a_0 to a_N respectively. The filter 120 also has an output portion 130 comprising a second series chain of latches 132_0 to 132_{M-1} with outputs connected to respective multipliers 134_1 to 134_M. The multipliers 134_1 to 134_M are associated with multiplicative coefficients b_1 to b_M respectively.

The outputs of both sets of multipliers 126_0 to 126_N and 134_1 to 134_M are summed by an adder 138. The filter 120 has an input 140 which is shown receiving x_n, this being the nth in a series x_1, x_2, x_3 ... etc. The latches 124_0 to 124_{N-1} and 132_0 to 132_{M-1} correspond to delays between computation stages, and are assumed to be equal. This is easily ensured in practice by providing additional internal delays for faster computation stages.

As illustrated, when multiplier 126_0 receives x_n, multiplier 126_1 is receiving x_{n-1} input one processing cycle earlier to multiplier 126_0 and delayed at latch 124_0. Similarly, the ith multiplier 126_i receives x_{n-i} ($i = 0$ to N). Simultaneously, the adder 138 has a sum output 142 providing y_n and connected to the first output latch 132_0. By virtue of delay at successive latches 132_0, 132_1, etc, the ith output multiplier 134_i receives y_{n-i} ($i = 0$ to M) when y_n is output by the adder 138. By inspection, it can be seen that y_n, the output of adder 138 is given by

$$y_n = a_o x_n + a_1 x_{n-1} + \ldots + a_N x_{n-N} + b_1 y_{n-1} + b_2 y_{n-1} + \ldots + b_M y_{n-M} \qquad (14)$$

$$\text{ie} \quad y_n = \sum_{i=0}^{N} a_i x_{n-i} + \sum_{i=1}^{M} b_i y_{n-i} \qquad (15)$$

The non-recursive summation term (FIR filter relation) involving x_{n-i} in Equation (15) may be computed by prior art multipliers. The recursive term (IIR filter relation) involving y_{n-i} is computed by building block processors 50 of the invention and illustrated in Figures 4 to 7. The invention accordingly provides for the general digital filter to be realised.

Referring now to Figure 10, there is shown a further embodiment 200 of a processor of the invention. The processor 200 operates in accordance with a radix 2 signed digit number representation arithmetic. It has a structure very similar to those of processors 10 and 50 described earlier, and is illustrated to indicate the consequences of a change of radix. Its description will accordingly be restricted to areas where it differs to previous embodiments. Figure 10 includes a box 202 illustrating each cell's input and output digit sets, together with a respective digital circuit for its implementation.

The processor 200 incorporates rows of multiplier cells such as third row cell 204, most of which consist of P, Q, R and T cells. For convenience, the distinction between cells and subcells will not be preserved. The third row begins with accumulator cells consisting of T and Q cells in cascade, an R cell and a T cell. The accumulator cells sum transfer digits from the most significant third row multiplier cell 204 with sum digits output from Q and R accumulator cells and the T cell of the most significant multiplier in the row immediately above (the second row).

Each P cell is arranged to multiply a respective digit b_1 by a feedback input digit y. Both digits are in the radix 2 redundant digit set (-1, 0, 1). The digit significance of b_1 ranges from $\overline{2}$ to $\overline{5}$. The multiplication produces a product in the range (-1, 0, 1). The product is passed to the associated Q cell, where it is added to an input digit in the range -1 to 1. The input digit is $u_{\overline{n}}^{\overline{m}}$ (m = 2 to 5) for top row multiplier cells or upper column neighbour sum outputs for second to fourth row multiplier cells. The Q cell produces a sum in the range (-2 to 2). This is represented by a sum digit w_{out} in the digit set (-2, -1, 0) and a transfer digit t equal to 0 or 1. The sum digit w_{out} is passed to the associated R cell, which adds it to a first transfer digit (equal to 0 or 1) from the adjacent multiplier cell. This produces a sum digit w_{out} equal to 0 or 1, which passes to the associated T cell for addition to a second transfer digit (equal to -1 or 0) from the adjacent multiplier cell. The multiplier T cell output is a single digit in the range -1 to 1, and passes down to the respective column neighbour cell below. The most significant multiplier cells such as 204 produce (as has been said) two transfer digits of value 0 or 1 and -1 or 0. These are added in the row neighbour T accumulator cell to produce a value in the range -1 to 1. This value is passed to the associated Q accumulator cell, which adds it to the column neighbour T cell output from above and in the range -1 to 1. The Q accumulator cell provides a transfer digit output equal to 0 or 1 to its row neighbour R accumulator cell. It also provides a sum output digit in the range -2 to 0 to its column neighbour R accumulator cell in the row below.

Each R accumulator cell sums the transfer digit (0,1) from its right with the sum output (-2, -1, 0) from above. It provides a transfer digit equal to -1 or 0 to its row neighbour T accumulator cell, together with a sum output digit equal to 0 or 1 to its column neighbour T accumulator in the row below. Each T accumulator cell receives transfer digits from above and to its right with respective values 0 or 1 and -1 or 0. It adds these to produce a single output digit in the range -1 to 1. This output digit becomes the row output, and is fed back as a multiplicand input to the respective immediately preceding row.

Figure 10 demonstrates that radix 2 processing increases the number of significance levels involved in accumulation as compared to earlier radix 4 processors; ie the processor 200 provides for an increase of up to two significance steps in accumulation as compared to one for the processor 10. However, since the processor 200 is employed with multiplicative coefficient digits b_1 of maximum significance $\overline{2}$ (as opposed to $\overline{1}$ for the processor 10), recycled digits are still fed back to the preceding row as in the earlier device.

The invention may be implemented with signed digit number representation arithmetic of any radix. Further details of such arithmetic are described by A Avizienis in "Signed-digit Number Representations for Fast Parallel Arithmetic" IRE Trans Computers Vol EC-10 pp 389-400, Sept 1961.

Referring now to Figure 11, this shows a version 220 of the Figure 10 radix 2 processor 200 with modifications in accordance with Figure 3; ie the processor 220 incorporates skewed inputs u_n^o etc and skewed outputs y_n^o etc. Coefficients b_i^o etc have maximum significance of zero. Top row multiplier cells contain only multiplication subcells (type P), there being no addition of terms such as v_n^o etc. These multiplier cells therefore do not generate transfer digits, although such a digit is generated in top row cell types Q and X. Multiplier cells in the second and subsequent rows do however generate transfer digits. Moreover, most significant multiplier cells each include a respective type Q subcell for addition of a non-recursive input digit u_n^o etc. This is analogous to the extra type B subcell in the most significant multiplier cells .in Figure 3.

CLAIMS

1. A recursive processor for multiplying successive recycled output results by a coefficient and adding their products to respective input terms, the processor including multiplier cells and accumulating means connected to form rows and columns, and wherein:

(1) each row is arranged to multiply a respective recycled digit by at least the more significant coefficient digits;
(2) the multiplier cells are associated with individual coefficient digits decreasing in significance along rows and increasing in significance down columns containing more than one such cell in each case;
(3) each multiplier cell is arranged to compute output sum and where necessary transfer digits corresponding to recycled and coefficient digits multiplied together and where necessary added to input digits;
(4) each row begins with a respective accumulating means having at least such of the following functions as are required by availability of relevant neighbouring processor elements:

(i) receipt of its most significant cell's transfer digit output,
(ii) receipt of output sum digits from the respective preceding row's accumulating means and most significant multiplier cell,
(iii) processing received digits to generate output digits of differing significance, passing the most significant of such digits to a respective processor output and passing the remaining one or more digits to accumulating means in a respective succeeding row;

(5) the multiplier cells and accumulating means operate in accordance with signed digit number representation arithmetic involving digit redundancy and providing most significant digit first computation;
(6) the processor includes row neighbour interconnection lines for passage of transfer digits between adjacent multiplier cells in the direction of increasing coefficient digit significance and between each row's most significant multiplier cell and accumulating means;

(7) the processor includes column neighbour interconnection lines arranged for sum digit transfer down each column between accumulating means, between multiplier cells and between most significant multiplier cells and accumulating means where available in each case, the column interconnection lines including clock-activated storing means for sum digit storage and advance between rows; and

(8) each accumulating means has a respective most significant digit output connectable as a recycled multiplicand digit input to all multiplier cells of a respective row selected in accordance with accumulation arithmetic, recycled digit significance and coefficient magnitude.

2. A processor according to Claim 1 wherein each first row multiplier cell is arranged to add a respective digit of a processor input term to the product of its associated coefficient digit and recycled multiplicand input digit.

3. A processor according to Claim 1 including most significant multiplier cells arranged to add respective processor input term digits to their respective multiplication products.

4. A processor according to Claim 3 arranged in cascade with a second like processor, and including recycle connections from the first processor's accumulating means most significant digit outputs to multiplicand inputs of respective rows of both processors, together with connections from the second processor's accumulating means most significant digit outputs to respective most significant multiplier cell addition inputs of the first processor.

5. A processor according to any preceding claim wherein the multiplier cells and accumulating means are arranged to operate in accordance with radix 4 or radix 2 arithmetic.

Amendments to the claims have been filed as follows

1. A processor for performing multiplication operations comprising multiplying input terms by a coefficient to form products, the processor (eg 10) including an array of multiplier cells (16) arranged to multiply by coefficient digits and to add product digits arising from multiplication, together with accumulating means (18,20,34) arranged to add array output digits, characterised in that the accumulating means (18,20,34) is arranged to compute output result digits most significant digit first and in descending order of digit significance in accordance with a redundant arithmetic scheme.

2. A processor according to Claim 1 wherein the multiplier cells (12) are connected to form rows (12) each arranged to multiply by at least the more significant digits, characterised in that each result digit is connected by a respective feedback line (30) to a respective multiplier cell row (12) as a common multiplicand input term digit for each multiplier cell (12) of the row (12) as appropriate to implement recursive computations.

3. A processor according to Claim 1 or 2 characterised in that the array is arranged to add respective second input terms to the products.

4. A processor according to Claim 1, 2 or 3 wherein the accumulating means (18,20,34) and multiplier cells (16) are connected to form rows (12) and columns (14) of the array, characterised in that:

(a) each row (12) containing multiplier cells (16) is arranged to multiply by at least the more significant coefficient digits with multiplier coefficient digit significance diminishing along the row;
(b) the columns (14) are associated with digits of respective significances, and columns (14) containing multiplier cells (16) exhibit multiplier coefficient digit significance increasing down the columns (14);
(c) the array is arranged to add second input terms to products by at least one of the following means:

(i) addition functions for first row multiplier cells,
(ii) addition functions for multiplier cells associated with most significant coefficient digits;

(d) intercell connections are arranged for movement down columns (14) of lower significance digits generated in the accumulating means (18,20,34) and multiplier cells (16) and for movement to neighbouring higher significance columns where available of like generated transfer digits;
(e) clock-activated latches (34) are arranged in intercell connections between column neighbour accumulating means parts (eg 18_{22},20_{32}) and multiplier cells (eg 16_{14},16_{24}) to provide for storage and advance of digits from row (12_{m}) to row (12_{m+1});
(f) multiplier cells (16) in at least the second and subsequent rows have addition functions for adding digits received from respective column neighbours to multiplication product digits and digits received from respective neighbouring columns where available; and
(g) the columns (14) terminate in respective accumulating means parts (18,20,34) arranged to generate respective output result digits in accordance with a redundant arithmetic scheme.

5. A processor according to Claim 1, 2, 3 or 4 characterised in that multiplier cells (16) required to add transfer digits received from respective neighbouring columns (14) are arranged to accept each such digit at a later computational stage to that in which it was generated as appropriate to restrict transfer digit propagation within any row (12).

6. A processor according to Claim 5 characterised in that multiplier cells (eg 316) associated with coefficient digits of lesser significance are connected to pass transfer digits between adjacent rows in accordance with a carry-save structure.

7. A processor according to Claim 6 characterised in that it includes multiplier cells (eg 304,316) arranged to receive multiplicand inputs in signed binary digit form and to perform addition in accordance with non-redundant binary arithmetic.

8. A processor according to Claim 3 characterised in that it is a first processor (50) arranged in cascade with a second like processor (50'), and in that the first processor (50) has accumulating means outputs connected to multiplicand inputs of both processors (50,50'), and the second processor (50') has accumulating means outputs connected to first processor additive inputs as appropriate for implementing second order recursive computations.

9. A processor according to Claim 3 characterised in that it is a member of a set of like processors (50,50',50") connected in cascade, and in that one processor (50) has accumulating means outputs connected to multiplicand inputs of the other processors (50',50"), and the other processors (50',50") each have accumulating means outputs connected to additive inputs of a respective succeeding processor (50',50).

Published 1990 at The Patent Office, State House, 66/71 High Holborn, London WC1R 4TP. Further copies may be obtained from The Patent Office Sales Branch, St Mary Cray, Orpington, Kent BR5 3RD. Printed by Multiplex techniques ltd, St Mary Cray, Kent, Con. 1/87